Planification de trajectoire pour drones de combat

Thibault Maillot

THÈSE DE DOCTORAT DE l'UNIVERSITÉ DU SUD TOULON VAR

Présentée par

Thibault MAILLOT

pour obtenir le grade de :

DOCTEUR DE L'UNIVERSITÉ DU SUD TOULON VAR
Mention AUTOMATIQUE

Équipe d'accueil : LSIS - Équipe projet ESCODI
École Doctorale : ED 184 - Mathématiques et Informatique de Marseille

Titre de la thèse :

Planification de trajectoires pour drones de combat

Soutenance prévue le 03/10/2013
devant le jury

Jean-Paul Gauthier *Professeur, Université de Toulon, LSIS*	Directeur de Thèse
Jean-François Balmat *Maître de conférences, Université de Toulon, LSIS*	Co-encadrant de Thèse
Ulysse Serres *Maître de conférences, Université Lyon 1, LAGEP*	Co-encadrant de Thèse
Boutaib Dahhou *Professeur, Université de Toulouse 3, LAAS-CNRS*	Rapporteur
Hassan Hammouri *Professeur, Université Lyon 1, LAGEP*	Rapporteur
Ugo Boscain *Directeur de recherche CNRS, École Polytechnique, CMAP*	Examinateur
Jean-Christophe Sarrazin *Ingénieur de recherche, ONERA/DCSD*	Invité
Jean-Alexis Tanchou *Directeur, OPERA ERGONOMIE*	Invité

Remerciements

Je tiens à remercier les personnes que j'ai rencontrées durant ces trois années de thèse et qui m'ont aidées aussi bien sur le plan scientifique que moral.

Je désire tout d'abord remercier mon directeur de recherche. Merci à Jean-Paul Gauthier de m'avoir accueilli dans son laboratoire pour travailler sur des problématiques de recherche que je ne connaissais pas. Je lui suis également reconnaissant pour le temps conséquent qu'il m'a accordé, ses qualités pédagogiques et scientifiques, sa franchise et sa sympathie. J'ai beaucoup appris à ses côtés. Le côtoyer pendant 3 ans fût un véritable plaisir.

J'adresse de chaleureux remerciements à mes co-encadrants de thèse, Jean-François Balmat et Ulysse Serres, pour leurs attentions de tout instant sur mes travaux, pour leurs conseils avisés et leurs écoutes qui ont été prépondérants pour la bonne réussite de cette thèse.

Mes remerciements au FUI pour avoir accepté de financer ce travail de thèse.

Un grand merci à Ugo Boscain pour son implication dans le projet, notamment sur la problématique de planification point-pattern. Il m'a beaucoup appris.

Je suis reconnaissant aux membres du jury, Jean-Christophe Sarrazin, Jean-Alexis Tanchou et particulièrement aux rapporteurs Boutaib Dahhou et Hassan Hammouri, pour l'intérêt qu'ils ont bien voulu porter à mon travail.

Je désire en outre remercier tous les membres de l'équipe ESCODI en plus de ceux précédemment cités, Alain Ajami, Nicolas Boizot, Francesca Chittaro, Nawal Daraoui, Jean Duplaix, Fredéric Lafont et Nathalie Pessel, pour leur sympathie, leur amitié. J'ai eu beaucoup de plaisir à travailler avec eux. Le cadre de travail était idéal.

Merci à Janie Azulay, Isabelle Lepagnay et plus particulièrement Valérie Minguy et Vincente Guis, je ne vous oublie pas. Merci pour votre bonne humeur.

Bien sûr, atteindre ces objectifs n'aurait pas été possible sans la gaîté de Yann Doh, Régis Abeille et Emilien Royer qui ont eu le grand plaisir de me supporter dans leur bureau.

Merci à ma famille qui m'a permis d'étudier dans de bonnes conditions.

Enfin, un grand merci à tous les autres collègues de bureau et personnes non cités, pour leur sympathie.

Table des matières

Table des figures

Nomenclature

$[\cdot,\cdot]$	Le crochet de Lie entre des champs de vecteurs et le commutateur des matrices
$\bar{\varepsilon_c}$ et σ_{ε_c}	Moyenne et écart-type des erreurs d'orientation de la caméra
$\bar{\lVert\varepsilon\rVert}$ et $\sigma_{\lVert\varepsilon\rVert}$	Moyenne et écart-type de la norme des erreurs de positionnement du vecteur
L	Le critère à optimiser
$\dot{x}$	Dérivée par rapport au temps de la fonction $x(\cdot)$
ε	Erreur de positionnement du vecteur par rapport à la cible
ε_c	Erreur d'orientation de la caméra
$\mathscr{H}$	Le Hamiltonien
$\langle a,b\rangle$	Le produit scalaire de a et b
$\mathcal{C}^{\mathbf{r}}$	Ensemble des fonctions f r-dérivables dont $f^{(r)}$ est continue
$\mathcal{Q}$	Variété de dimension n
q	Représentation d'état du drone
q_{cible}	Représentation d'état de la cible
$\mathscr{U}$	Ensemble des contrôles admissibles
$\rotatebox[origin=c]{180}{$\mathbf{P}_L$}$	Le problème de contrôle optimal inverse associé à $\mathbf{P}_L$
$\mathbf{P}_L$	Le problème de contrôle optimal minimisant L
drone HALE	drone volant à Haute Altitude et de Longue Endurance
drone MALE	drone volant à Moyenne Altitude et de Longue Endurance
TUAV	Tactical Unmanned Aircraft Vehicle
UAV	Unmanned Aircraft Vehicle
UCAV	Unmanned Combat Aircraft Vehicle

Introduction générale

Les véhicules aériens autonomes sont, de nos jours, des moyens de plus en plus utilisés. Ils sont aussi nommés drones, vecteurs ou UAVs. Le fait que le véhicule soit autonome (ou commandé à distance) permet d'effectuer des missions longues (la fatigue du pilote n'est pas un élément d'arrêt de mission) et en toute sécurité pour le pilote.

Ces véhicules autonomes sont classés en plusieurs catégories, en fonction de leurs missions respectives et de leurs possibilités de charges utiles (l'équipement pouvant être embarqué) [97, 123] :

- *Les drones tactiques (TUAV).* Ce sont des drones plutôt petits. Ils n'ont pas une portée d'action élevée et sont assez peu encombrants. L'autonomie de ce type de drone varie de quelques minutes à quelques heures.
- *Les drones volant à moyenne altitude et de longue endurance (MALE).* Ces drones permettent l'utilisation d'une charge utile pouvant atteindre 500 kg. Leur altitude de croisière est comprise entre 5 000 et 15 000 mètres. L'autonomie de ce type de drone peut être comprise entre 20 et 40 heures.
- *Les drones volant à haute altitude et de longue endurance (HALE).* L'altitude de croisière de ces drones est supérieure à 15 000 mètres.

Les drones de combat (UCAV) sont des engins pouvant embarquer une charge utile létale. Ils peuvent être tactique, MALE ou HALE.

Dans ce travail, nous ne nous intéresserons qu'aux drones HALE et MALE. De plus nous considérerons que la charge utile n'est autre qu'une caméra. En particulier, nous nous intéresserons aux problèmes de planification de trajectoires pour le système composé du vecteur et de sa charge utile.

0.1 Contexte

Le travail présenté dans ce document est le fruit d'une thèse financée par un FUI (Fonds Unique Interministériel), dirigée par le Professeur J.-P. Gauthier (Université du Sud-Toulon-Var) et co-dirigée par J.-F. Balmat (Université du Sud-Toulon-Var) et U. Serres (Université Claude Bernard Lyon 1).

Les drones constituent une technologie de plus en plus mise en avant dans différents projets [133, 77]. Le laboratoire LSIS se joint à cette recherche grâce à cette thèse subventionnée par le projet SHARE. Supporté par un consortium d'entreprises [1], le projet a pour objectif la création d'une station de contrôle au sol universelle pour des drones à voilure fixe ou tournante (e.g., des avions ou des hélicoptères).

Le but du laboratoire LSIS est d'assurer le couplage entre la trajectoire du drone (le vecteur) et ses capteurs (la charge utile). Ce couplage vecteur/capteur définit l'asservissement de la trajectoire du vecteur à la manipulation de la charge utile par l'opérateur, i.e. nous souhaitons conserver le champ de vision de l'opérateur en optimisant un des paramètres suivants :

- la discrétion,

1. Opéra Ergonomie, ONERA, Thales Alénia Space, Eurocopter, Adetel group

- l'angle de dépression de la caméra pour disposer de la meilleure observation (-45 à -30 degrés),
- l'orientation du soleil (éviter l'orientation face au soleil, éviter l'ombre portée du vecteur),
- la maximisation des capacités de zoom.

Dans certaines situations, le couplage vecteur/capteur asservira la manipulation de la caméra (et non celle du vecteur). C'est le cas, par exemple, lors de l'observation d'une zone prédéfinie via un modèle donné pour la caméra (e.g., nous souhaitons faire un état des lieux de tout un terrain).

Afin de satisfaire au mieux à cet objectif, nous devons prendre en compte les contraintes suivantes :

- Contraintes environnementales :
  - hauteur du sol,
  - sens du vent,
  - obstacles,
  - proximité du relief,
  - couloirs aériens imposés (Flight levels),
  - couloirs aériens interdits (No fly zone),
  - ...
- Contraintes sur le vecteur :
  - domaine de vol,
  - paramètres de vol (voilure fixe ou tournante),
  - paramètres du vecteur (ce que l'utilisateur peut commander),
  - ...
- Contraintes sur le capteur :
  - libertés de rotation,
  - vitesses de rotation,
  - temps de rafraîchissement,
  - ...

En plus des contraintes citées précédemment, le vecteur et les capteurs sont soumis à des modes de pilotage différents (choisis par l'utilisateur) et agissant à des niveaux de contrôle distincts [95, 94] :

- Modes de commandes du vecteur :
  - Mode de base : l'utilisateur commande le cap, la vitesse de vol, la vitesse d'ascension.
  - Mode automatique : l'utilisateur demande au vecteur de naviguer vers un point de passage défini.
  - Mode pattern : l'utilisateur demande au vecteur de suivre un *pattern* [94], i.e. une figure prédéfinie comme un cercle, un 8, un hippodrome...
- Modes de commandes du capteur :
  - Mode manuel : l'utilisateur commande le gisement, le site, le niveau de zoom.
  - Mode automatique 1 : l'utilisateur peut choisir un asservissement sur un point fixe.
  - Mode automatique 2 : l'utilisateur peut choisir un asservissement sur une cible mouvante.

Ainsi, toutes ces questions d'asservissement du vecteur et de la charge utile conduisent à une problématique de planification de trajectoires contraintes par les éléments cités précédemment. Cette planification de trajectoires doit permettre d'effectuer les missions demandées. Elles peuvent être :

- Observer un point fixe en utilisant un pattern autour du point d'intérêt à altitude constante, e.g., un bâtiment dont nous souhaitons connaître les caractéristiques,
- Observer un point fixe sans utiliser de pattern autour du point d'intérêt, e.g., une maison, une entrée d'une grotte ou d'une cache,
- Observer un point mobile dont la trajectoire est connue, e.g., un convoi ami à protéger,
- Observer un point mobile dont la trajectoire est inconnue, e.g., un convoi de chars ou de fantassins,
- Effectuer un balayage d'une zone d'intérêt fixe avec la caméra,
- Effectuer un balayage d'une surface mobile, i.e. effectuer une observation libre autour d'un point mobile, e.g., une protection de convoi.

Afin de satisfaire au mieux à ces missions, les algorithmes de planification devront permettre au vecteur et à la caméra de :

- Observer un point d'intérêt à 360 degrés,
- Observer un point d'intérêt sous un secteur angulaire (relatif à la cible),
- Observer un point d'intérêt sous un angle (azimut et élévation) souhaité. Ceci implique, entre autre, la capacité de spécifier l'angle d'observation par l'opérateur via l'interface utilisateur (IHM).

À terme, grâce au projet SHARE, l'utilisateur pourra effectuer simplement toutes sortes de missions, comme :

- Surveiller un objectif,
- Renseigner sur un objectif,
- Rechercher un objectif,
- Acquérir un objectif,
- Évaluer les dommages sur un objectif,
- Effectuer un relais de communication.

Remarque 0.1. Pour certaines missions, l'objectif peut être soit ami, soit ennemi.

Le but de ce travail est donc de construire des algorithmes de planification de trajectoires pour le vecteur permettant de prendre en compte la charge utile ainsi que les contraintes inhérentes au problème. Les algorithmes présentés par la suite sont voués à être utilisés en amont du mode automatique de commande, i.e. les algorithmes développés fournissent des points de passage à suivre par le vecteur et sa charge utile.

0.2 Contribution

Dans la suite, nous supposerons que la charge utile est une caméra positionnée sous le vecteur. De plus, nous considérons que le vecteur est soit un drone de type HALE (Haute Altitude Longue Endurance) volant à vitesse et altitude constante, soit un drone MALE (Moyenne Altitude Longue

Endurance) volant à vitesse constante.

Nous verrons que les équations du mouvement du drone HALE sont alors celles données par le système de la voiture de Dubins [2] [51], i.e.

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{\theta} = u, \end{cases}$$

avec $(x, y, \theta) \in \mathbb{R}^2 \times S^1$ l'état du système (où $(x, y) \in \mathbb{R}^2$ sont les coordonnées du drone dans le plan d'altitude constant, et θ le cap), et u est la variable de contrôle. Ce système a beaucoup été étudié, notamment dans [92, 116, 117].

Pour un drone de type MALE, nous verrons que les équations du mouvement peuvent être considérées comme celles de l'évolution de son repère cinématique [141, 71], i.e.

$$\begin{cases} \dot{X} = RV_0 \\ \dot{R} = R\Omega, \end{cases}$$

avec $(X, R) \in \mathbb{R}^3 \times SO(3)$ l'état du système où X représente les coordonnées du drone dans le repère global et R est la matrice de rotation entre le repère cinématique du drone et le repère global. $V_0 = (1, 0, 0)$ représente le vecteur vitesse unitaire dans le repère du drone et est dans l'alignement du corps de l'appareil. La matrice Ω représente la vitesse angulaire du solide considéré dans le repère cinématique du drone. C'est une matrice antisymétrique contenant les variables de contrôle u_1, u_2 et u_3 correspondant, respectivement, aux vitesses angulaires du roulis, tangage et lacet du drone.

Pour les modèles présentés ci-dessus, nous allons, dans un premier temps, proposer des méthodes de planification afin de répondre aux contraintes décrites au paragraphe précédent.

Dans un deuxième temps, nous répondrons à une problématique qui est naturellement apparue durant ce projet : comment proposer des trajectoires comparables à celles qu'un pilote expérimenté serait amené à décrire s'il était directement aux commandes de l'appareil ? L'objectif étant de se conformer au mieux aux règles de l'Organisation de l'Aviation Civile Internationale [74, circulaire 328, article 2.13].

2. Le système de Dubins est utilisé pour modéliser simplement une voiture. Nous l'utilisons pour notre vecteur en considérant une altitude fixe.

0.3 Présentation du manuscrit

Le manuscrit est organisé comme suit :

- La partie I propose des méthodes de planification et répond au problème de couplage vecteur/capteur :
  - Le chapitre 1 introduit le problème de planification.
  - L'état de l'art, la présentation des modèles choisis ainsi que des premières réponses au problème de planification sont proposées en chapitre 2.
  - Le chapitre 3 est une présentation de notre contribution avec les principaux résultats.
  - Le chapitre 4 dresse le bilan de la partie I.
- La partie II traite la problématique anthropomorphique. Dans cette partie, nous proposons une méthode, basée sur le contrôle optimal inverse, permettant au drone d'agir comme le ferait un pilote :
  - Nous introduisons cette problématique au chapitre 5.
  - Le chapitre 6 présente un état de l'art des méthodes bio-inspirées et énonce le problème de contrôle optimal inverse.
  - Les résultats obtenus, pour un drone HALE, sont présentés dans le chapitre 7.
  - Le chapitre 8 rappel les points importants de la partie II.

Première partie

Planification de trajectoires

Chapitre 1

Introduction

Au sein du projet SHARE, notre travail est d'assurer le couplage entre la trajectoire du drone (le vecteur) et ses capteurs (la charge utile). Ceci passe notamment par de la planification de trajectoires.

Dans cette partie, nous verrons quelles sont les méthodes actuellement utilisées pour répondre à ce type de problématique. Nous présenterons aussi celles que nous avons utilisées.

1.1 Le problème de planification

Le problème de planification soulevé a deux facettes. D'abord nous voulons que le drone se déplace selon une trajectoire d'un point à un autre de l'espace. Ensuite, nous voulons qu'en fonction de la trajectoire du drone, la charge utile embarquée remplisse sa mission.

Ce problème, aussi appelé couplage vecteur/capteur, implique l'utilisation de méthode de planification de trajectoires pour obtenir un chemin permettant de réaliser simultanément ces deux objectifs en ne compromettant pas les différentes contraintes inhérentes aux deux parties.

Le but est donc de trouver un (ou plusieurs) algorithme de planification pour que le vecteur et la charge utile atteignent leurs objectifs respectifs.

Par exemple, nous demandons au vecteur de se diriger vers une position fixe (en évitant les obstacles, ou les zones aériennes interdites) tout en demandant à la caméra embarquée de suivre une cible (sans la perdre de vue).

Le problème considéré peut être posé de la manière suivante :

Le problème de couplage vecteur/capteur

Étant donnée une position initiale du vecteur et de sa caméra (notée q_0), comment calculer une trajectoire $q(t)$, du drone et de sa caméra, définie sur $[0, T]$ (où T est un paramètre fixé ou libre) qui permette d'atteindre un point voulu q_{cible}.

La réponse à cette question sera donnée par la suite. Nous nous intéressons à deux cas de figure : un point final fixe (e.g., rejoindre un point de passage ou un pattern prédéfini) ou mobile (e.g., suivre un convoi).

Dans ces deux cas, il faut calculer une trajectoire de référence notée $q(t)$, définie sur $[0, T]$, satisfaisant les équations de la dynamique du vecteur et de la caméra, joignant les points q_0 et q_{cible} spécifiés tels que $q(0) = q_0$ et $q(T) = q_{\text{cible}}$.

La planification de trajectoires se fait en général en optimisant un critère : minimisation des efforts du moteur d'une voiture, de la consommation de carburant d'une fusée, maximisation du confort des passagers, minimisation de la longueur du chemin parcouru, minimisation du temps pour relier deux points...

De telles trajectoires optimisant un critère peuvent être obtenues grâce au Principe du Maximum Pontryagin, noté PMP [3]. Une application de cette méthode est présentée dans le paragraphe 2.2.3.

Dans de nombreuses applications, le PMP est utilisé en amont du développement pour trouver des morceaux de trajectoire (aussi appelés primitives) optimaux localement. La trajectoire sera alors simplement un recollement de primitives [70, 72].

Une autre méthode, très utilisée, consiste à discrétiser l'espace [57, 137, 96] et à appliquer des algorithmes de cheminement (basés sur la théorie des graphes), tels que les algorithmes de Dikjstra [1, 136], A^* [66, 140] ou d'autres encore [50].

D'autres manières de procéder existent, pour une présentation plus complète, voir [83].

Pour répondre au mieux à certains types de missions, la norme en vigueur [95] définit des trajectoires types, appelées patterns. Par exemple, lorsque le drone doit protéger un convoi, il doit suivre une trajectoire adaptée qui lui permet de surveiller le convoi et son voisinage proche.

Ces patterns sont aussi définis et utilisés dans les rapports [94, 72]. Les patterns souvent utilisés pour aider au bon déroulement des missions sont :

- Le cercle : ce pattern permet de faire de la surveillance d'objectif,
- L'hippodrome : ce pattern permet de protéger un convoi,
- Le huit : ce pattern permet de bombarder un objectif.

Lors de l'utilisation de patterns pour répondre aux besoins d'une mission, la question principale est de déterminer comment le vecteur peut le rejoindre. Après une étude des méthodes couramment utilisées, au paragraphe 2.3, nous en proposerons de plus élaborées au paragraphe 3.2.

Chaque méthode de planification présentée est développée dans le but d'être utilisée dans un module haut niveau, voué à être exécuté sur une machine distante. Celui-ci doit permettre de transmettre des points de passage à un module d'auto-pilotage du drone, voir Figure 1.1 ou, pour plus de détails, [4, Figure 5] (cf. Annexe B), [6, Figure 4] (cf. Annexe A), [25, Figure 3].

Cette architecture est définie par la norme STANAG 4586 [95] et elle est résumée dans l'article [46]. La méthode décrite dans [100] permet de créer un module bas niveau sur le drone, i.e. un auto-pilote. Dans [108], les auteurs utilisent un module de pilotage automatique du commerce.

Dans le cas d'une cible mobile, i.e. pour la poursuite de convoi, cette architecture peut être modifiée pour fournir toutes les données nécessaires à l'algorithme de planification.

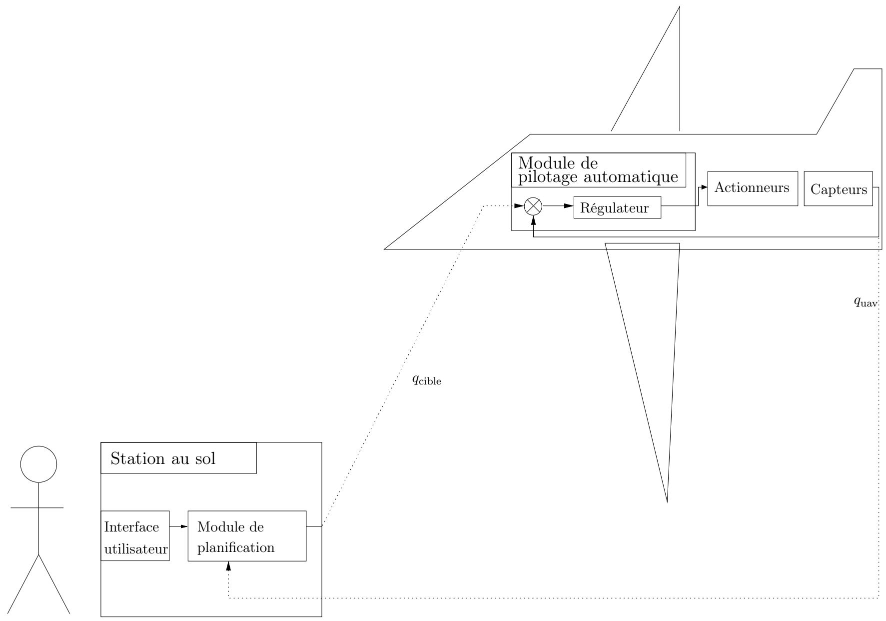


FIGURE 1.1 – Schéma de l'architecture considérée

En effet, si la cible est mobile, alors le point final q_{cible} est une fonction du temps. Cette fonction du temps doit être connue à chaque instant pour que l'algorithme puisse calculer une trajectoire. Quand la cible mobile est amie, cette information est supposée connue à chaque instant. Lorsque la cible est ennemie, ces données ne sont pas toujours connues.

Pour pallier ce manque d'information, nous devons réaliser une estimation de la trajectoire. Le schéma de l'architecture correspondant à ce cas de figure est représenté dans [25, Figure 3].

Rappelons que le problème de couplage vecteur/capteur doit aussi prendre en compte les contraintes énoncées en Introduction Générale. Nous devons mettre en place des méthodes permettant de ne pas entrer en conflit avec celles-ci.

1.2 Organisation de la partie I

Dans cette partie, nous traitons du problème de planification de trajectoires. Le chapitre 2 concerne l'état de l'art de la planification de trajectoires. Tout d'abord nous considérons le cas où la cible est un point fixe et dans un second temps la cible sera mobile.

Ensuite, dans le chapitre 3, nous présentons nos principaux résultats. Dans ce chapitre, le paragraphe 3.2 est le plus important : il décrit nos méthodes de planification vers un pattern donné.

L'étude du couplage vecteur/capteur est faite au paragraphe 3.1. Cette étude montre que le problème de couplage peut se restreindre à l'étude de la planification du vecteur seul.

Enfin, le paragraphe 3.4 propose une application des méthodes proposées pour effectuer du suivi de convoi en milieu encombré.

Chapitre 2

État de l'art

Sommaire

Dans ce chapitre, nous présentons quelques méthodes de planification de trajectoires pour un drone, dans le cas du suivi d'une cible fixe ou mobile.

Tout d'abord, nous abordons la modélisation des drones. La plupart des méthodes sont basées sur des modèles de comportement cinématiques des drones.

Pour répondre au problème de planification point-point, i.e. les points initiaux et finaux sont fixés, nous donnons ensuite un aperçu de méthodes de stabilisation et de contrôle.

Pour terminer, nous proposons des généralisations de ces méthodes pour effectuer du suivi de cible mobile. La cible est considérée comme amie ou ennemie, i.e. sa trajectoire future est connue ou non, et évolue dans un milieu encombré ou non.

Nous ne présentons ici que les méthodes de planification usuelles. Pour une bibliographie plus complète, nous vous renvoyons aux livres de LaValle [83] et Laumond [82]. Un aperçu des méthodes de suivi de convoi est présenté dans les articles [143, 42].

2.1 La modélisation

Nous allons présenter certains modèles cinématiques. Ils sont fréquemment utilisés en planification de trajectoires de véhicules évoluant dans un plan ou dans l'espace usuel à 3 dimensions.

Dans la suite nous considérons des systèmes de la forme générale

$$\dot{q} = f(q, u), \qquad q \in \mathcal{Q}, \quad u \in \mathscr{U} \tag{2.1}$$

où q est l'état du système et $\mathcal{Q}$ une variété lisse de dimension n. Le contrôle est donné par $u \in \mathscr{U}$ et $f(\cdot, u)$ est un champ de vecteur lisse, i.e. $\mathcal{C}^\infty$, sur la variété $\mathcal{Q}$.

Nous présentons les équations de chaque modèle, ainsi que les propriétés de contrôlabilité associées.

Dans la plupart des cas, la contrôlabilité est intuitivement évidente. Pour rappel, elle est définie comme suit [45] :

Définition 2.1 (Contrôlabilité d'un système)**.** Si, pour tout couple (q_0, q_{cible}) de deux états du système (2.1), il existe un temps T et une fonction localement intégrable $u : [0, T] \to \mathscr{U}$ telle que la solution du problème

$$\dot{q} = f(q, u), \qquad q(0) = q_0$$

satisfasse $q(T) = q_{\text{cible}}$, alors le système de contrôle est dit **contrôlable**.

Cette notion de contrôlabilité d'un système est une propriété importante. Par exemple, si le système est non contrôlable, la question d'existence de solutions optimales se pose.

Jean-Michel Coron, dans son cours [44], présente quelques méthodes d'analyse de la contrôlabilité. Il cite, pour les systèmes linéaires, le critère de Kalman qui donne une condition nécessaire et suffisante de contrôlabilité d'un système. Par contre, dans le cas de systèmes non-linéaires, la contrôlabilité n'est pas toujours évidente à prouver.

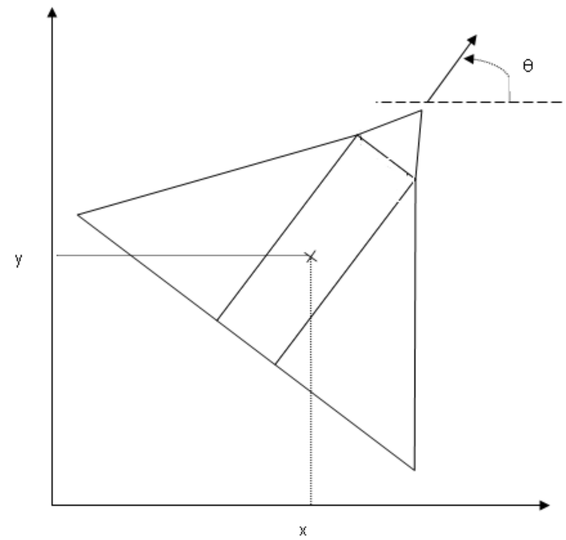


FIGURE 2.1 – Schéma d'un drone HALE

Pour les systèmes non-linéaires et sans drift, le théorème de Rashevski-Chow ([45, Théorème 2.3] ou [126]) donne une condition nécessaire de contrôlabilité partout locale, i.e. en temps petit il existe un contrôle qui permet d'atteindre tout point d'un certains voisinage du point de départ, sans en sortir.

2.1.1 Modélisation des drones HALE

Nous présentons une modélisation des drones de type HALE (Haute Altitude Longue Endurance). Puisque ces appareils volent à vitesse et altitude constantes, nous considérons l'étude de leurs mouvements dans leurs plans d'altitude fixe.

Un schéma d'une situation générique du drone HALE dans le plan est donné en Figure 2.1. Sur ce schéma, (x, y) définissent les coordonnées du centre de gravité du drone dans le plan d'altitude constante et θ définit son cap (la flèche représente le vecteur vitesse du système).

Un modèle largement utilisé pour ce type de véhicules est le système de Dubins [51, 16]

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{\theta} = u, \end{cases} \tag{2.2}$$

en notant $q = (x, y, \theta) \in \mathbb{R}^2 \times S^1$ l'état du système et u la variable de contrôle (généralement contrainte).

Ce système est de la forme générale (2.3), dite *affine dans le contrôle* :

$$\dot{q} = f(q) + ug(q), \tag{2.3}$$

avec

$$q = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix}, \quad f(q) = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix}, \quad g(q) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Remarque 2.2. La commande u appliquée au modèle de Dubins (2.2) représente la courbure de la trajectoire suivie par le système. En effet, rappelons que la courbure κ d'une courbe paramétrée dans un repère orthonormé $\gamma(t) = (x(t), y(t))$ s'exprime par

$$\kappa = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}}.$$

Le long d'une courbe du système (2.2), nous avons

$$\begin{aligned} \kappa &= \frac{\dot{\theta}\cos^2\theta + \dot{\theta}\sin^2\theta}{(\cos^2\theta + \sin^2\theta)^{3/2}} \\ &= u. \end{aligned}$$

Dans le cas des drones HALE, la commande u est contrainte par le rayon de courbure minimal du vecteur. L'ordre de grandeur de ce rayon de courbure est le kilomètre.

Remarque 2.3. Dans la modélisation (2.2), nous considérons que les drones HALE ont une vitesse normalisée égale à 1. Parfois (voir le paragraphe 3.2) nous considérons $u \in [-1, 1]$. Ces deux hypothèses sont possibles grâce à une normalisation du modèle cinématique du drone HALE.

En effet, si nous considérons que le drone HALE a une vitesse constante v et une commande vérifiant $u \in [-u_{\max}, u_{\max}]$, alors le système cinématique peut s'écrire sous la forme

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = u. \end{cases} \tag{2.4}$$

En utilisant la normalisation en temps et espace suivante

$$\begin{aligned} \tau &= 1/\iota_a t \\ \tilde{u} &= \iota_a u \\ \tilde{x} &= \iota_b x \\ \tilde{y} &= \iota_b y, \end{aligned}$$

avec $\iota_a = 1/u_{\max}$ et $\iota_b = u_{\max}/v$, le système normalisé vérifie (2.2) avec un contrôle compris dans l'intervalle $[-1, 1]$.

Un système similaire à (2.2) a été étudié par J.A. Reeds et L.A. Shepp [110]. Il autorise le véhicule à faire marche arrière, i.e. $v \in \{-1, 1\}$ dans le système (2.4).

La contrôlabilité des systèmes de Dubins et de Reeds-Shepp est en fait évidente. La raison théorique pour laquelle ces systèmes sont contrôlables est exposée dans le livre de Steven M. LaValle [83]. Pour une étude complète, se référer à [127].

2.1.2 Modélisation des drones MALE

Le paragraphe 3.2 traite de la problématique de la planification dans le cas d'un drone MALE (Moyenne Altitude Longue Endurance) qui évolue dans l'espace à 3 dimensions.

Dans ce cas, le modèle considéré doit représenter les principales caractéristiques d'un drone MALE, i.e. nous souhaitons pouvoir modifier sont cap ainsi que son altitude. Nous supposons de nouveau qu'il a une vitesse constante unitaire.

Considérons un système de Dubins (similaire au drone HALE) que nous étendons en associant une commande indépendante sur l'altitude, comme suit

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{z} = u_z \\ \dot{\theta} = u_\theta, \end{cases} \tag{2.5}$$

avec $q = (x, y, z, \theta) \in \mathbb{R}^3 \times S^1$, l'état du système (où $(x, y, z) \in \mathbb{R}^3$ sont les coordonnées du drone dans l'espace, et θ son cap). Les variables de contrôle sont alors u_θ, pour commander le cap, et u_z, pour modifier l'altitude.

Le modèle ainsi formé est appelé modèle de type avion de Dubins et a été utilisé de nombreuses fois dans certains problèmes de planification [40, 94, 60].

Ce modèle ne représente pas l'évolution du drone en tant que corps solide dans $\mathbb{R}^3$.

Certains modèles de drones MALE d'un niveau de complexité plus élevé sont présentés dans [26, 58, 62].

Si nous restons au niveau purement cinématique, les équations classiques de la mécanique du solide constituent une représentation raisonnable d'un drone MALE. Ces équations constituent d'ailleurs la base des modèles plus précis [26, 58, 62].

Nous considérons donc l'évolution du repère de l'avion (le repère cinématique) par rapport au repère global (galiléen, nous supposons la terre plate et immobile).

L'état q associé au drone est composé de la matrice de passage du repère cinématique au repère global ainsi que des coordonnées $X = (x_1, x_2, x_3)$ du centre de masse dans le repère global (cf. Figure 2.2).

En reprenant les notations de la Figure 2.2, l'état peut s'écrire sous la forme $q = (X, R)$ avec la matrice de rotation R vérifiant

$$\begin{bmatrix} \vec{i} & \vec{j} & \vec{k} \end{bmatrix} = R \begin{bmatrix} \vec{i}_{\text{uav}} & \vec{j}_{\text{uav}} & \vec{k}_{\text{uav}} \end{bmatrix}.$$

Nous supposons que le repère cinématique est contrôlé en vitesse angulaire par trois commandes correspondant aux commandes du roulis, tangage et lacet du drone. Pour plus de détails, se référer à [26, p.43].

Au final, le système cinématique peut s'écrire sous la forme [141, 71]

$$\begin{cases} \dot{X} = RV_0 \\ \dot{R} = R\Omega \end{cases} \tag{2.6}$$

avec

- $q = (X, R) \in \mathbb{R}^3 \times SO(3)$, l'état du système où X représente les coordonnées du drone dans le repère global et R est la matrice de rotation entre le repère cinématique du drone et le repère global,
- $V_0 = (1, 0, 0)$, le vecteur vitesse normalisé à 1 dans le repère du drone, qui se trouve dans l'alignement du corps de l'appareil,
- $\Omega = \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix} \in \mathfrak{so}(3)$, la vitesse angulaire du solide considérée dans le repère cinématique du drone ($\mathfrak{so}(3)$ est l'algèbre de Lie de $SO(3)$, constituée des matrices antisymétriques).

Les variables de contrôle u_1, u_2 et u_3 correspondent, respectivement, aux vitesses angulaires du roulis, tangage et lacet.

La contrôlabilité du modèle (2.6) a été étudiée d'un point de vue mathématique dans [119] et est obtenue en appliquant [115, Théorème 6.7] (ou bien [28, Théorème 1]). Elle est encore intuitivement évidente.

Remarque 2.4. En ne considérant que des rotations autour de l'axe $\vec{k}$ et des commandes de roulis et tangage nulles, l'évolution des trajectoires du système (2.6) n'est rien d'autre que celle du système de Dubins (2.2) dans un plan horizontal d'altitude fixée dès que la vitesse initiale de l'avion est dans un plan horizontal.

2.2 Planification point-point

Disposant maintenant de représentations mathématiques des véhicules considérés, nous sommes en mesure de traiter le problème de planification. Nous allons, dans cette partie, étudier quelques méthodes applicables dans le cas d'une cible fixe. Nous appelons ce problème la planification de trajectoires point-point.

De nombreuses méthodes sont utilisées pour répondre à ce problème. Citons, par exemple, les travaux de Jean *et al.* [75, 38, 88] basés sur la théorie du contrôle ou celui de Takei *et al.* [128] qui est basé sur une résolution numérique des équations de Hamilton-Jacobi. Pour une bibliographie plus étendue, se référer à [83].

2.2.1 La platitude

Une première méthode utilisable pour répondre à ce problème de planification de trajectoires est l'utilisation de la propriété de platitude des systèmes.

La notion de platitude a été développée par M. Fliess *et al.* au début des années 1990 [55]. Elle est présentée dans un article des Techniques de l'ingénieur [111], inspiré du livre [90], de la manière suivante :

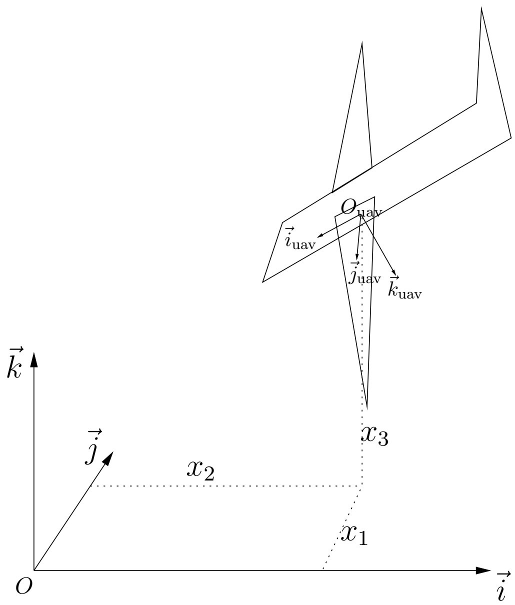


FIGURE 2.2 – Schéma du drone en 3D

> "Cette propriété, qui permet de paramétrer de façon très simple le comportement dynamique d'un système, est basée sur la mise en évidence d'un ensemble de variables fondamentales du système : ses sorties plates. [...]
> Une sortie plate est composée d'un ensemble de variables qui permet de paramétrer toutes les autres variables du système, l'état, la commande, mais également la sortie."

M. Fliess, dans [54], définit la platitude comme suit

Définition 2.5 (Platitude d'un système)**.** La platitude est une propriété d'un système d'équations différentielles ordinaires, sous-déterminé, c'est-à-dire à moins d'équations que d'inconnues. Le système est dit **plat** si il existe m variables $\mathbf{y} = (y_1, \ldots, y_m)$ telles que :

1. Toute variable du système s'exprime comme fonction différentielle de $\mathbf{y}$, i.e. une fonction des composantes de $\mathbf{y}$ et de leurs dérivées jusqu'à un ordre fini.
2. Toute composante de $\mathbf{y}$ s'exprime comme fonction différentielle des variables du système.
3. Les composantes de $\mathbf{y}$ et leurs dérivées sont linéairement indépendantes.

$\mathbf{y}$ est appelé sortie plate ou linéarisante.

L'intérêt de la platitude est qu'elle permet une forme de linéarisation exacte du système, ce qui permet de se ramener à un problème classique de commande linéaire.

Dans le domaine de l'ingénierie, la commande par platitude est surtout utilisée pour les problèmes de poursuite de trajectoires. M. Fliess *et al.* présentent de nombreux exemples d'applications dans les articles [55, 54, 53]. Cette méthode a été reprise pour commander l'entrée dans l'atmosphère d'une navette spatiale [93] et pour réaliser des fonctions de suivi de trajectoires en temps réel [138].

Plus récemment, elle a été utilisée pour mimer la préhension des rapaces par un quadcopter [131].

Bien que cette méthode soit utilisée assez fréquemment, nous ne l'appliquerons pas pour faire de la planification de trajectoires. Par contre, nous pouvons, avec cette méthode de contrôle, mettre en œuvre une loi de guidage pour effectuer du suivi de trajectoire, i.e. s'en servir dans le module de pilotage automatique. Dans ce cas de figure, une méthode de planification doit être utilisée pour fournir des points de passage, comme proposé dans [131, Figure 7].

2.2.2 La méthode de Lyapunov : une utilisation du principe de LaSalle

La méthode de Lyapunov est une méthode de stabilisation vers un point ou un ensemble de points choisis (e.g., un pattern).

Avant d'expliquer cette méthode, rappelons la définition de la stabilité.

Définition 2.6 (Stabilité au sens de Lyapunov [102])**.** En reprenant les notations précédemment définies, considérons un système différentiel de la forme $\dot{q} = f(q),\ q \in \mathcal{Q}$.
Supposons qu'il existe un point d'équilibre $\bar{q} \in \mathcal{Q}$, i.e. $f(\bar{q}) = 0$.
L'équilibre $\bar{q} \in \mathcal{Q}$ est dit **stable** (au sens de Lyapunov) si et seulement si pour tout $\epsilon > 0$, il existe $\eta > 0$ tel que pour toute condition initiale q_0 vérifiant $\|q_0 - \bar{q}\| \leq \eta$, la solution de $\dot{q} = f(q)$ issue de q_0 à $t = 0$, est définie pour tout temps t positif et vérifie $\|q(t) - \bar{q}\| \leq \epsilon$.
Si le système n'est pas stable, il est dit instable.

En plus de la définition de la stabilité autour d'un point d'équilibre, nous définissons la stabilité asymptotique et la stabilité globalement asymptotique, pertinente dans notre cas.

Définition 2.7 (Stabilité asymptotique [102])**.** En reprenant les notations de la Définition 2.6, l'équilibre $\bar{q}$ est dit localement **asymptotiquement stable** si et seulement si il est stable et si, de plus, il existe $\eta > 0$ tel que toutes les solutions $q(t)$ de $\dot{q} = f(q)$, partant en $t = 0$ de conditions initiales q_0 telle que $\|q_0 - \bar{q}\| \leq \eta$, convergent vers $\bar{q}$ lorsque t tend vers $+\infty$.

Lorsque la condition initiale peut être librement choisie, $\bar{q}$ est dit **globalement asymptotiquement stable**.

Le théorème de Lyapunov ou le principe d'invariance de LaSalle [80], qui est une extension de celui-ci, sont appliqués pour prouver la stabilité globalement asymptotique d'un système sans contrôle $\dot{q} = f(q)$. Ils sont énoncés dans [102, 135] et d'une manière plus concise dans [61, Théorème 3.5, p.190]. Nous les rappelons ci-après.

Définition 2.8 (Fonction de Lyapunov [135])**.** Nous reprenons les notations de la Définition 2.6.
Soit Ω un ouvert de $\mathbb{R}^n$ contenant le point d'équilibre $\bar{q}$. La fonction $V : \Omega \to \mathbb{R}$ est une **fonction de Lyapunov** en $\bar{q}$ sur Ω si

- V est $\mathcal{C}^1$ sur Ω,
- $V(\bar{q}) = 0$ et pour tout $q \in \Omega \setminus \{\bar{q}\},\ V(q) > 0$,

- V décroît le long des trajectoires de $\dot{q} = f(q)$, i.e. pour tout $q \in \Omega$,

$$\dot{V}(q) = \langle \nabla V(q), f(q) \rangle = \sum_{i=1}^{n} \frac{\partial V}{\partial q_i}(q) f_i(q) \leq 0$$

Si, pour tout $q \in \Omega \setminus \{\bar{q}\}$, $\dot{V}(q) < 0$, la fonction de Lyapunov est dite stricte.

Théorème 2.9 (Théorème de Lyapunov [135]). *Nous reprenons les notations de la Définition 2.8. Si il existe une fonction de Lyapunov au point d'équilibre $\bar{q}$ sur Ω, alors le point $\bar{q}$ est stable. Si la fonction de Lyapunov est stricte alors $\bar{q}$ est asymptotiquement stable.*

Théorème 2.10 (Principe d'invariance de LaSalle [135]). *Nous reprenons les notations de la définition 2.8.*
Soit $V : \Omega \to \mathbb{R}^+$ une fonction de classe $\mathcal{C}^1$ telle que

- *V est propre sur Ω, i.e. $\forall T \in V(\Omega)$, $V^{-1}([0, T])$ est compact dans Ω,*
- *$\forall q \in \Omega, \quad \langle \nabla V(q), f(q) \rangle \leq 0$.*

Soit Γ le plus grand sous-ensemble de $\{q \in \Omega \mid \langle \nabla V(q), f(q) \rangle = 0\}$ invariant par le flot (en temps $t \geq 0$) de $\dot{q} = f(q)$. Alors toute solution $q(t)$ de $\dot{q} = f(q)$ tend vers Γ.

Remarque 2.11. Dans le Théorème 2.10, si Γ se réduit à un point d'équilibre $\bar{q}$ alors $\bar{q}$ est globalement asymptotiquement stable dans Ω.

P. Rouchon *et al.* expliquent [102, Partie 3.5.2] comment utiliser ces résultats pour stabiliser le système (2.3).

La méthode, dite de *Lyapunov*, consiste à synthétiser, en se servant du Théorème 2.9 ou du Théorème 2.10, une loi de commande qui conduit à la stabilité asymptotique globale.

Cette méthode est utilisée dans de nombreuses applications. Par exemple, certains auteurs appliquent cette méthode pour éviter les collisions entre les drones [47] ou bien pour commander des robots bio-inspirés [36].

Le problème de suivi de convoi sera aussi traité par cette méthode, cf. paragraphe 2.3.4.

Nous nous inspirons de cette méthode au paragraphe 3.2 pour le problème de planification point-pattern (i.e. le drone doit rejoindre un ensemble de points).

La méthode de Backstepping

Une autre méthode de stabilisation, présentée dans [76, Paragraphe 14.3], est construite à partir du principe de LaSalle. Elle s'applique à une certaine classe de systèmes non-linéaires à un contrôle, appelée *strict-feedback form* par l'auteur, et donnée par

$$\begin{cases} \dot{x} = f_0(x) + g_0(x)z_1 \\ \dot{z}_1 = f_1(x,z_1) + g_1(x,z_1)z_2 \\ \dot{z}_2 = f_2(x,z_1,z_2) + g_2(x,z_1,z_2)z_3 \\ \quad \vdots \\ \dot{z}_{k-1} = f_{k-1}(x,z_1,\ldots,z_{k-1}) + g_{k-1}(x,z_1,\ldots,z_{k-1})z_k \\ \dot{z}_k = f_k(x,z_1,\ldots,z_k) + g_k(x,z_1,\ldots,z_k)u \end{cases}$$

où $x \in \mathbb{R}^n$ et z_1 à z_k sont des scalaires et $u \in \mathbb{R}$ le contrôle. De plus, dans cette classe de systèmes, nous supposons que les f_i, $i \in \{0,\ldots,k\}$, s'annulent à l'origine et que $g_i(x,z_1,\ldots,z_i) \neq 0$, $i \in \{0,\ldots,k\}$ sur tout le domaine.

En utilisant itérativement le principe de LaSalle, nous construisons une commande u qui permet de stabiliser ces types de systèmes. Cette méthode de stabilisation est dite de *backstepping*.

Une utilisation de la méthode de *backstepping* sur un système simple est proposée dans [102, Paragraphe 3.5.2]. Dans cet exemple, en reprenant les notations précédentes, l'auteur considère $k = 1$ et $\dot{z}_1 = u$.

2.2.3 Le problème de contrôle optimal

Les méthodes de contrôle optimal sont utilisées pour piloter les drones tout en minimisant une certaine fonction de coût (appelée critère), notée J. Ce critère à optimiser peut, par exemple, être le temps de parcours ou la longueur de la trajectoire.

L'outil classique pour résoudre ce type de problème est le Principe du Maximum de Pontryagin, noté PMP [3]. Nombreuses sont les utilisations de ce principe. La mécanique hamiltonienne en est un exemple où le critère à optimiser n'est autre que l'action, i.e. l'intégrale sur le temps du Lagrangien. Dans ce cas, ce principe est dit d'action minimale et la quantité de mouvement fait office de ce que nous nommerons, par la suite, vecteur adjoint [2].

Dans la suite, nous noterons $(\mathbf{P}_L)$ le problème optimal associé à une fonction de coût L et nous le définissons de la manière suivante :

Problème de contrôle optimal $(\mathbf{P}_L)$

Minimiser le coût intégral $J(q,u) = \int_0^T L(q(t),u(t))dt$ parmi tous les contrôles admissibles $u(\cdot)$ qui permettent, sous la contrainte du système (2.1), de relier un point initial q_0 à un point final q_{cible} en un temps T, libre ou fixé.

Remarque 2.12. Un point q_{cible} étant donné, l'ensemble des extrémales solutions du problème $(\mathbf{P}_L)$ à partir de tout point q_0 est appelé *synthèse* optimale du problème $(\mathbf{P}_L)$.

Pour répondre au problème ($\mathbf{P}_L$), il faut d'abord étudier l'existence des ses solutions. Généralement, l'utilisation du théorème de Fillipov [3, Théorème 10.1] permet de prouver un tel résultat.

Ensuite, les trajectoires optimales sont obtenues en utilisant le Principe du Maximum de Pontryagin, dont voici l'énoncé [19]

Théorème 2.13 (Principe du Maximum de Pontryagin (PMP))**.** *Considérons le système de contrôle de la forme* (2.1) *et un coût de la forme* $J(q,u) = \int_0^T L(q(t),u(t))dt$ *où* L *est une fonction lisse par rapport à* q*. Le problème est de relier un point initial* q_0 *à un point final* q_{cible} *en un temps* T *fixé. Si le couple* $(q(\cdot),u(\cdot))$ *est optimal sur* $[0,T]$*, alors il existe une application* $p(\cdot) : [0,T] \to \mathbb{R}^n$ *absolument continue appelée vecteur adjoint, et un réel* $\lambda \leq 0$*, tels que, pour tout* $t \in [0,T]$ *le couple* $(\lambda, p(t))$ *est non trivial, et tels que, pour presque tout* $t \in [0,T]$*,*

$$\begin{aligned} \dot{q}(t) &= \frac{\partial \mathscr{H}}{\partial p}(\lambda, p(t), q(t), u(t)), \\ \dot{p}(t) &= -\frac{\partial \mathscr{H}}{\partial q}(\lambda, p(t), q(t), u(t)), \end{aligned} \tag{2.7}$$

où $\mathscr{H}(\lambda,p,q,u) = \langle p, f(q,u)\rangle + \lambda L(q,u)$ *est le Hamiltonien du système, et de plus, nous avons la condition de maximisation presque partout sur* $[0,T]$

$$\mathscr{H}(\lambda, p(t), q(t), u(t)) = \max_{v\in\mathscr{U}} \mathscr{H}(\lambda, p(t), q(t), v) = Cste \tag{2.8}$$

Remarque 2.14.

- Si de plus le temps final T pour rejoindre la cible q_{cible} n'est pas fixé, nous avons la condition suivante : $\mathscr{H}(\lambda, p(t), q(t), u(t)) = 0, \ \forall t \in [0,T]$.
- Il existe des versions plus générales du PMP [3, 139].
- Ce théorème est une condition nécessaire d'optimalité.
- Si le contrôle n'est pas borné ou qu'il se trouve à l'intérieur du domaine $\mathscr{U}$ des contrôles admissibles, alors l'équation (2.8) entraîne $\frac{\partial \mathscr{H}}{\partial u}(\lambda,p,q,u) = 0$.
- Définissons une *trajectoire extrémale* comme un quadruplet $(\lambda, p(\cdot), q(\cdot), u(\cdot))$ qui vérifie les conditions nécessaires d'optimalité du PMP.
  Si $\lambda = 0$, l'extrémale est dite *anormale*.
  Si $\lambda < 0$, l'extrémale est dite *régulière*. Dans ce cas, le vecteur adjoint étant défini à un scalaire multiplicatif près, nous pouvons choisir les valeurs usuelles $\lambda = -1$ ou $\lambda = -1/2$.
- Si les points q_0, q_{cible} ne sont pas fixés mais appartiennent à des variétés lisses $\mathcal{Q}_0$ et $\mathcal{Q}_1$ de dimensions finies, il faut rajouter les conditions de transversalité :

  $$\begin{aligned} p(0) &\perp T_{q(0)}\mathcal{Q}_0 \\ p(T) &\perp T_{q(T)}\mathcal{Q}_1 \end{aligned}$$

  L'idée sous-jacente est que, si le point final ou de départ est libre, alors parmi tous les problèmes point-point possibles, il faut choisir celui (ou ceux) qui minimise(nt) le coût.

À titre d'exemple, nous allons appliquer le PMP sur un cas simple. Nous étudierons les trajectoires du système de Dubins (2.2) qui minimisent la fonctionnelle J représentant un compromis pondéré entre la longueur et l'énergie du signal de contrôle $u(\cdot)$ d'une trajectoire $q : [0,T] \to \mathcal{Q}$.

Celle-ci est définie par

$$J(q,u) = \int_0^T 1 + \alpha u(t)^2 dt,$$

avec $\alpha > 0$ une constante de pondération.

Les trajectoires minimisant un critère similaire ont été étudiées par Yuri Sachkov *et al.* dans une série d'articles [92, 116, 117] pour un système de type Reeds-Shepp.

Écrivons le problème $(\mathbf{P}_{L_\alpha})$, pour le système de Dubins, associé au coût $L_\alpha(u) = 1 + \alpha u^2$:

$(\mathbf{P}_{L_\alpha})$ Minimisation d'un coût intégral

$$J(u) = \int_0^T L_\alpha(u(t))dt$$

par rapport à tous les contrôles admissibles $u(\cdot) \in \mathbb{L}^\infty([0,T],\mathscr{U})$ conduisant le système (2.2) d'un point de départ q_0 à un point d'arrivée q_{cible} en un temps T libre.

Soit le Hamiltonien

$$\mathscr{H}(\lambda,p,q,u) = p_x\cos(\theta) + p_y\sin(\theta) + p_\theta u + \lambda L_\alpha(u), \tag{2.9}$$

avec $p = (p_x, p_y, p_\theta) \in \mathbb{R}^3$ et $\lambda \leq 0$.

Si $(\lambda, p(\cdot), q(\cdot), u(\cdot))$ une extrémale du problème $(\mathbf{P}_{L_\alpha})$ alors elle vérifie les conditions du PMP, dont le système (2.7).

Comme le temps final T est libre, le Hamiltonien est identiquement nul le long de la trajectoire optimale [3], i.e.

$$0 = p_x\cos\theta + p_y\sin\theta + p_\theta u + \lambda L_\alpha(u). \tag{2.10}$$

L'équation de q dans (2.7) n'est rien d'autre que (2.2), quand à l'équation de p, aussi appelée *équation adjointe*, elle peut explicitement être écrite

$$\begin{cases} \dot{p}_x = 0 \\ \dot{p}_y = 0 \\ \dot{p}_\theta = p_x\sin\theta - p_y\cos\theta \end{cases} \tag{2.11}$$

Remarque 2.15. Par la suite, les valeurs constantes de $p_x(\cdot)$ et $p_y(\cdot)$ seront notées, respectivement, p_x et p_y.

En utilisant les équations (2.2), (2.8), (2.10) et (2.11) nous sélectionnons les trajectoires candidates à être solution de $(\mathbf{P}_{L_\alpha})$.

Dans le cas où une synthèse complète du problème considéré a été obtenue (comme par exemple dans l'article [117] pour le système de Reeds-Shepp avec un critère similaire à celui présenté précédemment), il suffit d'appliquer les règles obtenues lors de la synthèse pour construire la trajectoire optimale.

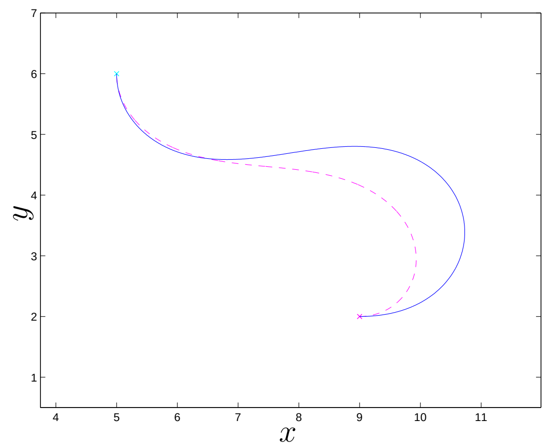


FIGURE 2.3 – Résultats de planification optimale, utilisant le PMP, minimisant le compromis L_α en temps libre (la trajectoire continue correspond à $\alpha = 10$ et la trajectoire en pointillé à $\alpha = 2$). Le point initial est $q_0 = (9, 2, 0)$ et le point final est $q_{\text{cible}} = (5, 6, \pi/2)$.

Si ce n'est pas le cas, il faut calculer la synthèse numériquement en intégrant le système hamiltonien (2.7). La trajectoire solution est construite en cherchant la valeur du vecteur adjoint initial (dans notre cas $p(0) = (p_x, p_y, p_\theta(0)))$ et du temps final T, s'il est libre. Si le temps est fixé, l'équation (2.10) n'est plus valable.

Les algorithmes que nous utilisons pour obtenir numériquement la synthèse sont des algorithmes de tir. Pour plus d'informations, voir [121, 135, 93, 83].

Deux solutions du problème ($\mathbf{P}_{L_\alpha}$) sont représentés en Figure 2.3.

Dans ces expériences, deux points arbitrairement fixés sont reliés par une trajectoire. Le temps est libre et le contrôle du cap n'est pas contraint, i.e. $u \in \mathbb{R}$. La différence entre les expériences est la valeur de pondération du compromis utilisée. Nous constatons que pour obtenir la trajectoire continue, où la valeur de α est la plus grande, l'énergie du contrôle est moins importante (la trajectoire à une courbure moins importante).

Beaucoup de travaux ont été effectués pour produire des synthèses de problèmes optimaux minimisants le temps. En premier lieu, nous pouvons citer le travail de U. Boscain et B. Piccoli qui, dans leur livre [31], exposent une méthode de construction des synthèses optimales en temps pour des systèmes de dimension 2. Cet ouvrage a comme point de départ les résultats de [32, 33, 103, 125].

Les premiers travaux sur les trajectoires temps-minimal pour les systèmes de Dubins et de Reeds-Shepp ont été effectués, respectivement, par L.E. Dubins [51] et Reeds et Shepp [110]. Ils ont caractérisé les trajectoires optimales en temps avec un contrôle borné.

Les trajectoires temps-minimal du système de Reeds-Shepp ont été étudiées par Sussmann et Tang [127] ainsi que Boissonnat *et al.* [27]. Souères et Laumond ont construit la synthèse temps-minimal de ce système [122].

La synthèse temps-minimal pour le système de Dubins a été obtenue par Bui, Boissonnat, Souères et Laumond [34]. Souères *et al.* étudièrent aussi les trajectoires optimales en temps pour rejoindre une droite et ont produit la synthèse associée [14]. Dans le même temps, Bakolas donna une synthèse temps-minimal pour le système de Dubins en présence de vent [13].

Bien d'autres problèmes de planification ont été étudiés grâce à ce principe [35].

Dans le paragraphe 3.2, nous allons calculer la synthèse du problème temps-minimal pour rejoindre un cercle, pour le système de Dubins.

2.3 Le suivi de convoi

Dans cette partie, nous nous intéressons au problème de planification de trajectoires dans le cas d'un suivi de cible mobile. La problématique est donc d'utiliser les méthodes présentées précédemment pour prendre en compte la non stationnarité du point final.

La planification de trajectoires vers un point final en mouvement permet de répondre à certains besoins spécifiques des missions effectuées par un drone (e.g., espionnage, protection). Pour cela nous pouvons aussi utiliser les patterns définis par la norme mise en place par l'organisation de l'aviation civile internationale [95].

Nous ne nous intéressons qu'au contrôleur de haut niveau du vecteur. En d'autres termes, nous supposons qu'il existe un module embarqué qui permet au vecteur de suivre une trajectoire prédéfinie (par points de passage par exemple). C'est le cas dans [108, 25] où la trajectoire fournie par un module de planification est transmise à un contrôleur de vol du commerce, qui effectue le contrôle de bas niveau (cf. Figure 1.1).

Différentes méthodes de suivi de cibles sont décrites dans [143, 42].

Classiquement, les auteurs utilisent des méthodes stochastiques [22], des méthodes d'optimisation numérique [78] ou encore des méthodes basées sur un modèle de type proie/prédateur [12].

2.3.1 Méthode quadratique

Dans [29], les auteurs présentent une méthode quadratique de contrôle qui permet de prendre en compte les contraintes sur l'état et la commande. Nous l'adaptons à la planification de trajectoires.

Nous supposons que le système est affine dans le contrôle, i.e. il est de la forme (2.3). La méthode consiste à trouver un contrôle minimisant l'écart entre la vitesse $\dot{q}$ et la valeur cible $\dot{q}_{\text{cible}}$.

Il faut alors chercher à exprimer cet écart en fonction de la commande u. Le but est de résoudre le problème de minimisation quadratique associé, avec ou sans contraintes sur l'état et/ou sur la commande [105].

La commande $u^*(\cdot)$ solution du problème de minimisation permettant de construire la trajectoire peut répondre, par exemple, au problème de la forme (cf. paragraphe 3.1)

$$(Z - Ru^*)^T \Delta (Z - Ru^*) = \min_{u \in \mathscr{U}} \left[(Z - Ru)^T \Delta (Z - Ru) \right],$$

où Z, R sont des matrices et Δ est une matrice de pondération.

Ce problème de minimisation est standard et peut être résolu très simplement grâce à la programmation quadratique (e.g., par une méthode de relaxation [105]).

Dans le paragraphe 3.1, nous utiliserons cette méthode quadratique pour répondre au problème de couplage vecteur/capteur.

2.3.2 L'utilisation de primitives

Le problème de planification, en temps réel, est soumis à de nombreuses contraintes, dont celle du temps de calcul. C'est pourquoi, beaucoup de modules de planification sont basés sur des trajectoires types, ou primitives, que peut effectuer le vecteur. La trajectoire planifiée est alors composée d'une concaténation de trajectoires types.

Ces primitives peuvent être construites à partir de la résolution d'un problème de contrôle optimal. Par exemple, Latombe a utilisé le modèle de Reeds-Shepp, et ses courbes temps-minimal, pour construire un algorithme de planification rapide pour une voiture autonome, dans un milieu encombré [81].

Frazzoli *et al.* ont proposé des trajectoires types, appelées *trim primitives*, pour un hélicoptère à 6 degrés de libertés. À partir de ces primitives de mouvement, obtenues par le biais d'un module de pilotage automatique, ils construisent un module de planification concaténant ces morceaux de trajectoires pour optimiser un certain coût [58].

Cette approche a été utilisée pour la planification de trajectoires de robots bipèdes. Par exemple, Hauser *et al.* proposent une méthode probabiliste de planification utilisant des primitives de mouvement d'un robot humanoïde [68].

Enfin, dans [85, 70], les auteurs utilisent des primitives de trajectoires définies à partir de la différence de vitesse entre le vecteur et sa cible.

2.3.3 Des méthodes basées sur des considérations géométriques

Pour certaines missions, des patterns prédéfinis sont proposés et, dans ce cas, des algorithmes de planification temps-réel peuvent être mis en œuvre.

Une première méthode, décrite dans Rafi *et al.* [107], consiste à donner un cap à suivre au vecteur permettant de rejoindre un cercle (à la même altitude que le vecteur) centrée sur la cible considérée. C'est la méthode de la tangente qui n'est utilisable que si le vecteur est à l'extérieur du cercle visé. Dans le cas où le vecteur est à l'intérieur du cercle, une autre méthode de planification est appliquée, par exemple une méthode de stabilisation basée sur le principe de LaSalle [37].

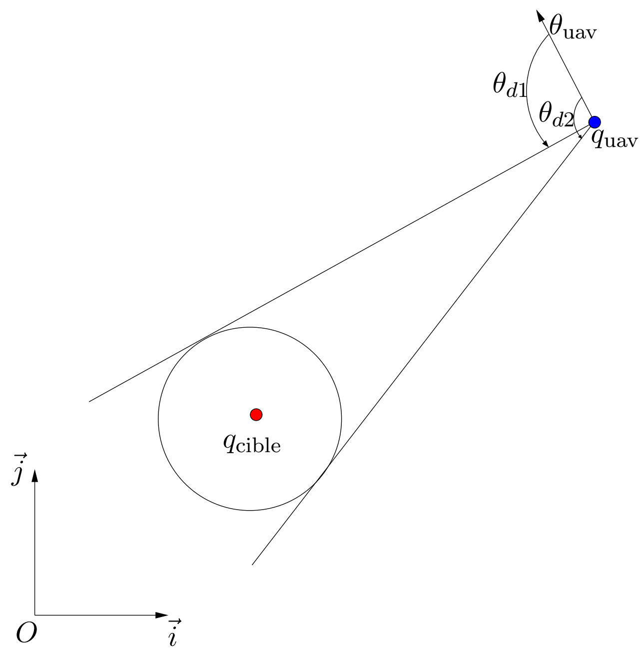


FIGURE 2.4 – Schéma du principe de la méthode de la tangente

Lorsque le vecteur est en dehors du cercle, l'idée est de considérer les deux tangentes au cercle, passant par le centre de masse du drone. Le cap $\theta_{\text{uav}_{\text{cible}}}$ que doit suivre le vecteur est celui qui est le plus proche du cap actuel et qui permet de suivre une des deux tangentes. En utilisant les notations de la Figure 2.4, $\theta_{\text{uav}_{\text{cible}}} = \theta_{\text{uav}} + \min(\theta_{d1}, \theta_{d2})$, avec θ_{d1}, θ_{d2} les angles entre le cap actuel et les tangentes au cercle. Cette commande de cap est envoyée au module de pilotage automatique du drone.

Une autre méthode, basée sur des considérations géométriques, est développée dans l'article de Theodorakopoulos *et al.* [130]. La problématique concerne les drones HALE et le couplage vecteur/capteur. Le vecteur doit rejoindre la projection du point cible dans son plan d'altitude constante avec une trajectoire maximisant le temps d'observation de la cible. La caméra embarquée est soumise à des contraintes sur son champ de vision.

Le cap que doit prendre le drone est calculé de sorte que :

- La cible reste au centre du champ de vision de la caméra,
- La distance entre le vecteur et la cible soit plus grande que le rayon de courbure minimal.

La loi de guidage latéral permet de contrôler le cap du vecteur.

Après une étude géométrique du problème, les auteurs constatent qu'une trajectoire en spirale permet d'obtenir un temps de visibilité de la cible maximal.

2.3.4 Les méthodes utilisant le principe de LaSalle

Comme mentionné au paragraphe précédent, certains auteurs utilisent des méthodes basées sur le principe de LaSalle pour stabiliser le drone sur un pattern.

Wood, dans sa thèse [142], ainsi que Hamel *et al.* , dans [63, 89, 71, 21], ont considéré une méthode de suivi de trajectoire, basée sur le principe de LaSalle, pour un véhicule à décollage vertical.

Lawrence *et al.* ont appliqué de telles techniques sur des modèles de drone de type HALE [84, 59, 60].

Ce principe de LaSalle est aussi utilisé pour trouver des méthodes de suivi de cible décrivant des trajectoires types (e.g., des droites ou des cercles) [100, 144].

Dans tous les travaux cités, les méthodes de stabilisation sont appliquées en supposant que le problème de suivi concerne la stabilisation dans le plan d'altitude constante du drone. Nous allons, dans le paragraphe 3.2, proposer une méthode, basée sur le principe de LaSalle, permettant de passer outre cette supposition et susceptible d'être appliquée à des drones de type MALE.

2.3.5 La résolution du problème de suivi via le PMP

Le Principe du Maximum de Pontryagin a déjà été exploité pour la mise en œuvre d'algorithmes de suivi de convoi.

Par exemple, les trajectoires en temps-minimal, pour le système de Dubins, sont utilisées par Bhatia *et al.* [25, 24]. Dans ces articles, les auteurs considèrent plusieurs drones et veulent qu'ils rejoignent le même point, au même instant. Cet exemple peut être étendu, par la suite, pour effectuer de la reconnaissance en coopération.

Ce même problème a été étudié, plus récemment, par Ortiz *et al.* dans [98], pour faire de la surveillance de zone à plusieurs drones.

Ding *et al.* ont considéré le problème de la protection d'un convoi en utilisant des trajectoires temps-minimal, du système de Dubins, rejoignant un cercle centré sur le convoi [48]. Dans cet article, le rayon de courbure minimal des trajectoires du drone est supposé plus grand que le rayon du cercle de protection autour de la cible si bien que le vecteur ne peut pas suivre le pattern autour du convoi.

Dans le paragraphe 3.2, nous proposerons une synthèse des trajectoires temps-minimal, pour un drone de type HALE, permettant de rejoindre le cercle de rayon le rayon de courbure minimal du vecteur. Cette synthèse peut être utilisée pour la protection des convois.

2.4 Prédiction du mouvement de la cible

Dans la partie précédente nous avons proposé quelques méthodes pour rejoindre une cible. Dans le cas d'une mission de protection d'un convoi (une cible amie), la position cible q_{cible} est considérée connue (le convoi peut communiquer ses coordonnées au drone). Cependant, dans le cas d'une cible ennemie, les méthodes ont besoin d'un module supplémentaire qui permet d'obtenir les informations nécessaires.

La plupart des auteurs, pour estimer la position de la cible, utilisent des méthodes probabilistes [42, 86].

Dans cette partie, nous présentons, succinctement, deux méthodes couramment utilisées.

2.4.1 Le filtre de Kalman

Une méthode classique d'estimation de la position future de la cible est basée sur un filtre de Kalman étendu. Ce filtre estime les états d'un système dynamique à partir d'une série de mesures incomplètes et bruitées. Le filtre de Kalman étendu est utilisé pour observer des systèmes non-linéaires : c'est une extension du filtre de Kalman linéaire basée sur la linéarisation autour de la trajectoire estimée.

Ce filtre est présenté, par exemple, dans le livre de Besançon [23] et est appliqué sur un quadcopter dans l'article de Boizot *et al.* [120].

Remarque 2.16. Le filtre de Kalman étendu ne converge pas globalement. Il existe, par contre, des preuves de convergence locale [15].

Cette méthode de prédiction de la position d'une cible est utilisée, par exemple, dans l'article de Prévost *et al.* [106]. Nous l'utilisons dans le paragraphe 3.2 pour réaliser du suivi de convoi.

2.4.2 Le filtrage particulaire

Le filtrage particulaire, ou méthode de Monte-Carlo séquentielle, est une technique d'estimation de modèles basée sur des simulations probabilistes.

Cette méthode est détaillée dans [83, 132, 86, 129]. Les principales étapes sont les suivantes :

1. Initialisation : fixer un certains nombres de particules p, correspondant à des états initiaux distribués uniformément dans l'espace d'état et calculer leurs poids associés (en fonction de leurs vraisemblances),
2. Propagation : calculer les mouvements de chaque particule en fonction d'un modèle probabiliste [49], puis calculer leurs poids associés,
3. Sélection : tirer avec remise p particules, dans l'ensemble des particules existantes, proportionnellement à leurs poids. Associer le poids $1/p$ aux p nouvelles particules. Retourner à l'étape 2.

Le modèle de mouvement Brownien est un exemple de modèle appliqué à l'étape 2.

Le filtrage particulaire a été utilisé par Hongda *et al.* dans [37] pour estimer la position d'une cible mobile.

2.5 L'évitement d'obstacles

Toutes les méthodes proposées jusqu'à présent ont été développées dans le but d'être utilisées dans un environnement non obstrué, i.e. dans le cas où le vecteur peut se déplacer dans tout l'espace.

Dans la réalité, l'environnement est souvent obstrué. Dans ce paragraphe, nous verrons comment détecter des obstacles puis planifier la trajectoire dans un environnement contraint.

Le terme obstacle désigne, ici, aussi bien des entités physiques (e.g., bâtiments, reliefs, ...) que virtuelles (e.g., zones de non vol, frontière, ...).

Afin d'éviter ces obstacles, il est nécessaire de les localiser. Si l'emplacement des obstacles virtuels est supposé connu, ce n'est pas le cas pour les obstacles physiques. En effet, à tout moment, un objet peut entrer dans l'environnement proche du vecteur.

Il est important de modéliser ces obstacles. Nous souhaitons créer une entité permettant de définir un obstacle (physique ou virtuel) pouvant être reconnu par le module de planification.

Enfin, une fois les obstacles définis, il faut utiliser les méthodes de planification connues afin de proposer une trajectoire réalisant nos objectifs (e.g., les objectifs du couplage vecteur/capteur) et évitant d'entrer en collision avec les obstacles.

Ces trois points seront traités, succinctement, dans cette partie. Pour une bibliographie plus complète, se référer à [83].

2.5.1 La détection des obstacles

Nous allons ici énoncer quelques méthodes, couramment utilisées, pour construire un module de détection d'obstacles physiques.

Le plus naturellement possible, pour mimer le comportement humain, la vision peut être exploitée pour construire un module de détection. En effet, dans la vie courante, l'homme utilise principalement sa vision pour se repérer et repérer les obstacles proches.

Kim *et al.* font de la reconnaissance de scène grâce à de la vision. Dans une série d'articles [79, 109, 59], ils ont mis en place, sur un drone, un module de détection et de suivi de routes via une caméra embarquée.

Par ailleurs, Kim *et al.* ont implanté sur un drone un module de recherche et de suivi de rivière, basé sur du traitement d'images [108].

Pour obtenir de meilleurs résultats, la caméra peut être couplée à un radar. Ce couplage de capteurs est appliqué à la détection d'objets et à la reconstruction de scènes [124].

Au lieu d'un radar, certains travaux utilisent un laser pour permettre une détection sur un environnement plus large. C'est le cas, par exemple, dans [101] où l'auteur présente une caméra couplée à un dispositif laser pour effectuer de la détection d'obstacles, pour un problème d'assistance à la conduite.

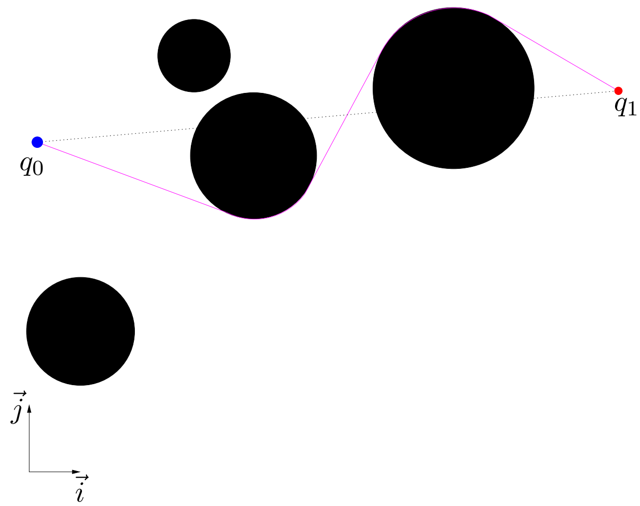


FIGURE 2.5 – Schéma d'une méthode simple d'évitement d'obstacles. La trajectoire relie les points q_0 et q_1 et évite les obstacles (en noir).

Nous ne développerons pas plus ce paragraphe puisque, par la suite, nous supposerons que la position des obstacles est connue. Pour plus de détails sur le sujet, se référer à [64].

2.5.2 La discrétisation de l'espace

En considérant la position des obstacles connue (physiques ou virtuels), il faut les modéliser et les prendre en compte dans l'algorithme de planification.

En appliquant une méthode simple de planification, comme la méthode de la tangente, les obstacles sont modélisables par des cercles [37]. Ces cercles doivent englober les obstacles et avoir un rayon plus grand que le rayon de courbure minimal du drone. Le cap objectif à envoyer au module de pilotage automatique est celui qui permet de suivre les tangentes aux obstacles situés entre le point cible et le vecteur (courbe continue sur la Figure 2.5).

Le problème principal de cette méthode est le suivant : si l'environnement du drone est constitué d'un couloir étroit, alors nous aboutissons à l'obstruction du couloir. Par exemple, sur la Figure 2.6 l'algorithme de planification considérera que le drone ne peut pas passer entre les deux obstacles puisque leurs cercles circonscrits se chevauchent au niveau du couloir.

Ce problème d'obstruction peut aussi survenir en considérant d'autres formes géométriques simples pour modéliser les obstacles (e.g., des carrés ou des triangles).

Une méthode plus efficace de modélisation des obstacles passe par une discrétisation plus fine de l'espace. L'idée est alors de considérer des formes géométriques, que nous appelons cellules, permettant de découper l'environnement. Les obstacles sont alors constitués d'un ensemble de cellules. Une discrétisation simple peut être effectuée par des carrés, e.g., les pixels de l'image de la Figure 2.7.

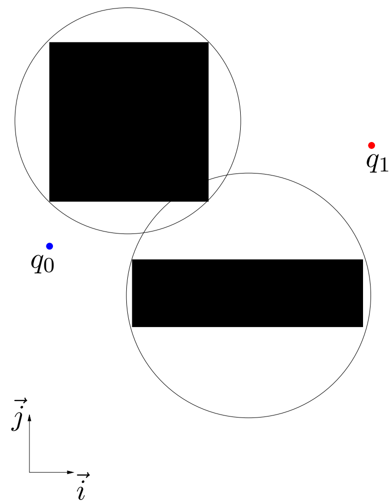


FIGURE 2.6 – Schéma d'une configuration d'obstacles ne convenant pas à la modélisation par des formes géométriques simples.

La décomposition par cellules identiques, i.e. l'environnement est discrétisé uniformément par une forme simple, peut poser problème. En effet, il peut y avoir une majorité d'éléments ne contenant pas d'informations utiles (e.g., les pixels blancs de la Figure 2.7). Une idée serait alors de les regrouper pour constituer des cellules plus grandes (pas nécessairement identiques) pour modéliser l'espace. La décomposition de Boustrophédon [50, 91] le permet.

Cependant, il existe une méthode de discrétisation plus efficace. Celle-ci est basée sur la triangulation de Delaunay et les graphes de Voronoï [137, 96]. Elle utilise la théorie de Morse discrète introduite dans [57]. L'idée est d'utiliser les graphes de Voronoï afin de créer une cellule par obstacle, la frontière de ces cellules étant toujours équidistante de tous les obstacles. Une stratégie de guidage sécuritaire est de suivre cette frontière. Un exemple d'utilisation de cette discrétisation est présenté en Figure 2.8.

Cette triangulation a été utilisée, par exemple, par Belta *et al.* dans [18]. Nous l'utiliserons dans la prochaine partie ainsi que dans le paragraphe 3.4.

2.5.3 Les méthodes de planification en milieu contraint

Une fois une modélisation des obstacles obtenue, par discrétisation de l'espace par exemple, nous appliquons une méthode de planification.

Généralement, après cette discrétisation de l'espace, des algorithmes de planification discrets sont appliqués. La majeure partie de ceux-ci sont basés sur la théorie des graphes [73].

Certains auteurs planifient le mouvement du vecteur en combinant les éléments d'un alphabet de trajectoires [18]. D'autres utilisent des algorithmes de cheminement, plus communément appelés

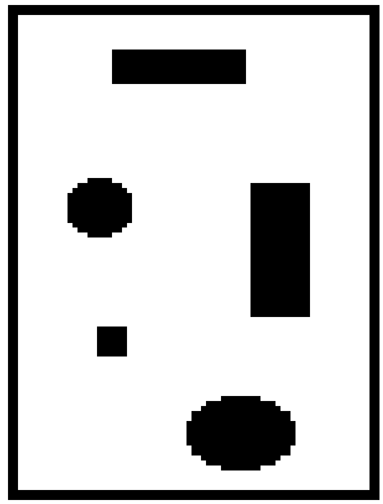


FIGURE 2.7 – Schéma d'un environnement obstrué, discrétisé via une pixelisation.

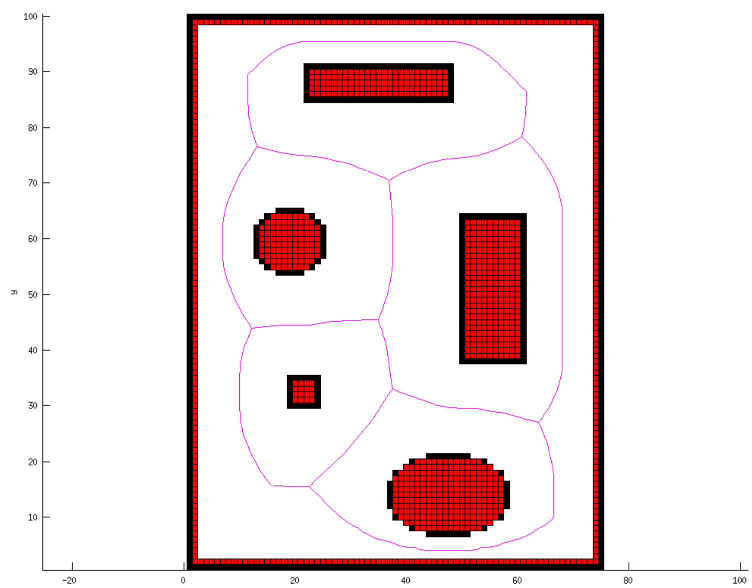


FIGURE 2.8 – Exemple d'un graphe de Voronoï.

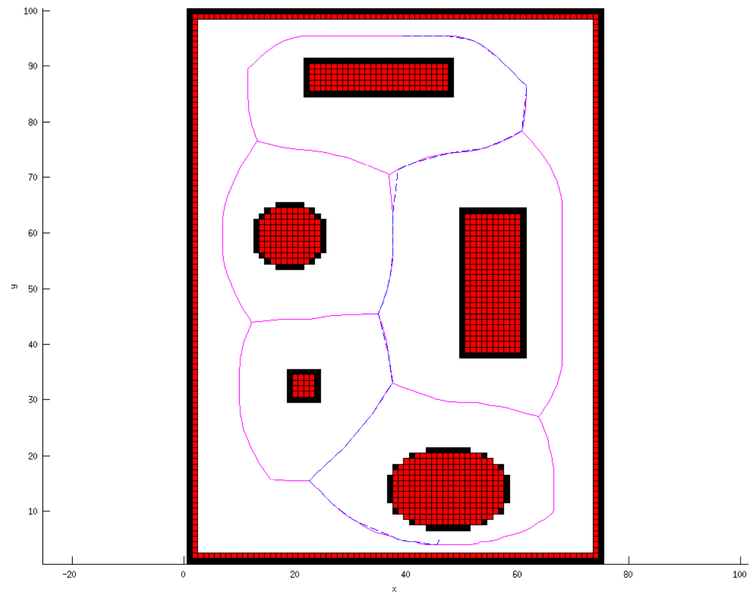


FIGURE 2.9 – Exemple d'un fil d'Arianne (en pointillés bleu).

algorithmes de déménageur de piano [50, 73], tels que l'algorithme de Dikjstra [1, 136] ou le A^* [66, 140].

Un exemple d'utilisation de l'algorithme A^* est proposé dans [69]. Des points de passage sont calculés via cet algorithme et sont fournis au contrôleur bas niveau du drone, basé dans ce cas sur un PID. Un exemple de trajectoire fournie par cet algorithme est représenté sur la Figure 2.9. Sur cette figure, le résultat de la planification est représenté en pointillés. La discrétisation est construite à partir du graphe de Voronoï, cf. Figure 2.8.

La méthode présentée précédemment, utilisant des algorithmes de cheminement, permet de calculer une trajectoire évitant les obstacles. Cependant les trajectoires calculées ne sont pas toujours admissibles pour le vecteur car sa dynamique ne permet pas de les suivre.

Une idée serait alors d'appliquer le PMP avec contraintes sur l'état, sur le système cinématique du drone [135, 43, 104]. Cette méthode permettrait d'obtenir une trajectoire pouvant être suivie par le drone. De plus, en considérant des contraintes sur l'état, i.e. des ensembles auxquels l'état ne doit pas appartenir, la trajectoire évitera aussi les obstacles.

Bien que l'idée d'obtenir une trajectoire optimale soit intéressante, le PMP contraint est assez difficile à mettre en œuvre. Afin d'utiliser les avantages du PMP (l'optimalité des trajectoires au sens d'un certain coût et l'utilisation de contraintes dynamiques), une méthode a été mise en œuvre par Latombe dans [81]. La planification se fait en deux étapes :

1. La première étape consiste à calculer une trajectoire, grâce à un algorithme de cheminement, reliant les points de départ et d'arrivée, tout en évitant les obstacles. Nous nommerons ce

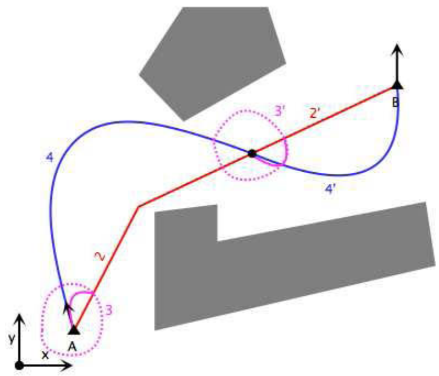


FIGURE 2.10 – Principe de l'algorithme d'évitement d'obstacles

chemin le *fil d'Ariane* qui peut être, par exemple, le résultat de l'algorithme A^* en Figure 2.9.

2. La deuxième étape consiste à utiliser ce fil d'Ariane comme support pour calculer une trajectoire admissible pour le véhicule en utilisant, par exemple, le PMP ou des primitives de trajectoires optimales en temps.

En résumé, l'idée sous-jacente est de réduire le problème contraint à des sous-problèmes sans contrainte sur l'état.

Nous avons utilisé cette idée pour construire un algorithme d'évitement d'obstacles.

Tout d'abord, cet algorithme doit trouver des points de passages dans le but d'appliquer une méthode de planification sans contrainte et ainsi obtenir une trajectoire admissible. La recherche de ces points se fait grâce à l'algorithme discret A^* qui permet de trouver un fil d'Ariane, reliant les points de départ et d'arrivée, et respectant les contraintes d'état.

Le PMP est alors appliqué pour trouver une trajectoire optimale, en fonction d'un coût choisi par l'utilisateur, reliant la position courante du vecteur à un point du fil d'Ariane.

Voici les étapes de construction de la trajectoire. Celle-ci est illustrée sur la Figure 2.10.

Étape 1 Initialisation du point courant (la position du vecteur),

Étape 2 Recherche du fil d'Ariane, par discrétisation de l'espace et en utilisant l'algorithme A^* (courbes annotées 2 et 2' sur la Figure 2.10),

Étape 3 Recherche d'une trajectoire, avec le PMP, reliant le point courant et un point du fil d'Ariane, le plus éloigné possible. Les courbes annotées 3, 3' et 4, 4' de la Figure 2.10 représentent cette étape[1],

Étape 4 Mise à jour du point courant et retour à l'étape 2, tant que celui-ci n'est pas proche du point d'arrivée (sur la Figure 2.10, cette boucle est effectuée deux fois).

1. Les courbes annotées 3, 3' correspondent à un prétraitement effectué localement pour améliorer la recherche des courbes optimales (annotée 4, 4').

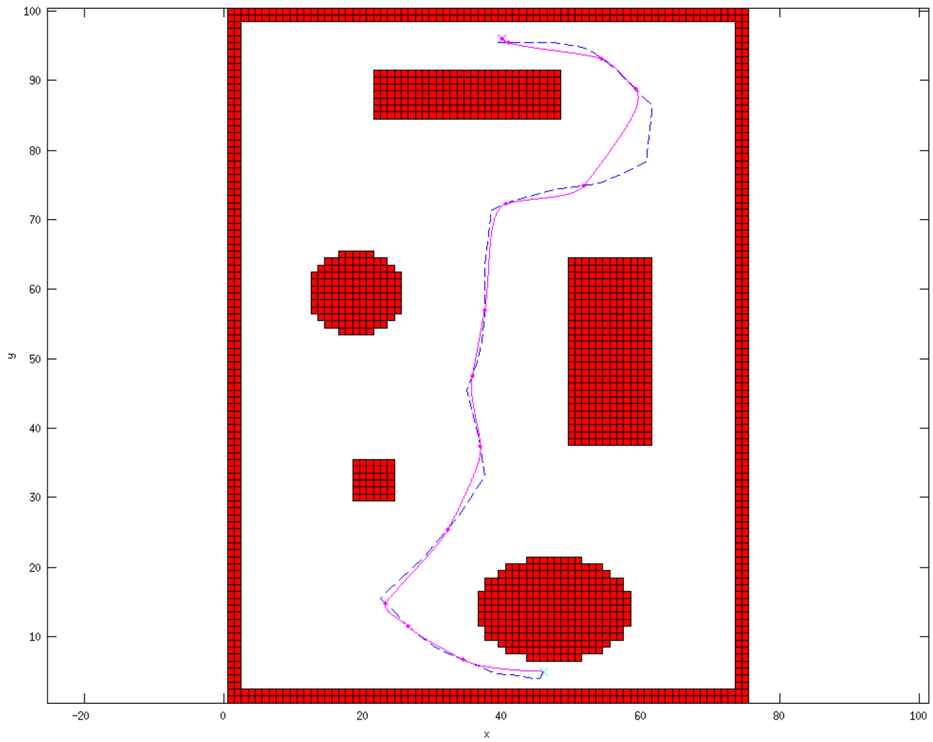


FIGURE 2.11 – Recherche d'une trajectoire optimale, en terme de longueur et de courbure, avec un temps final libre dans un espace contraint.

Cette méthode de recherche de trajectoires a été implantée sous Matlab. La Figure 2.11 correspond à une planification vers un point final fixe, en temps libre. La courbe en pointillés représente la trajectoire non-admissible utilisée comme fil d'Ariane pour calculer une trajectoire admissible et évitant les obstacles (courbe continue).

Chapitre 3

Résumé des contributions personnelles

Sommaire

Jusqu'ici nous n'avons présenté qu'un panorama des méthodes de planification de trajectoires. Nous traitons maintenant le problème du couplage vecteur/capteur.

Au paragraphe 3.1, nous montrons que le problème du couplage vecteur/capteur peut être résolu simplement en cas de contraintes en position sur la caméra. Dans cette même partie, nous proposons une méthode permettant de prendre en compte la caméra lorsque celle-ci n'est soumise à aucune contrainte de mouvement ce qui nous amènera à constater que le problème du couplage se situe essentiellement dans la planification du vecteur (pour éviter de perdre de vue la cible à cause d'un mauvais positionnement).

Les questions qui se dégagent sont les suivantes :

1. Comment effectuer le couplage vecteur/capteur ?
2. Comment définir une trajectoire permettant de rejoindre un pattern nous positionnant dans des conditions optimales pour l'observation de la cible ?
3. Comment suivre une cible mobile pour l'observer au mieux ?
4. Comment suivre une cible mobile en milieu contraint (e.g., en présence d'obstacles) ?

Dans la suite, nous donnons une réponse à ces problématiques. Les réponses aux questions 2 et 3 sont détaillées dans les articles présentés au paragraphe 3.2 et dans les Annexes A et B. Ces articles sont publiés ou en cours de publication.

3.1 Comment effectuer le couplage vecteur/capteur ?

Dans ce paragraphe, nous proposons une stratégie de couplage vecteur/capteur dans le cas d'un point final fixe ou mobile.

Cette stratégie de contrôle est présentée au paragraphe 3.1.1. Elle induit une première pré-étude du problème de couplage vecteur/capteur (cf. paragraphe 3.1.2) :

- Prise en compte des contraintes en position de la caméra pour effectuer la planification du vecteur.
- Dans le cas où la caméra n'est pas contrainte, le problème de couplage revient à étudier uniquement la planification du vecteur (cf. paragraphe 3.2). Dans ce cas, nous suggérons une méthode indépendante de contrôle de la caméra (cf. paragraphe 3.1.3).

3.1.1 Description de la méthode

L'algorithme utilisé pour répondre au problème de planification posé au paragraphe 1.1 est inspirée de [29] (introduit au paragraphe 2.3.1). Celui-ci vise à trouver une trajectoire qui permet de prendre en compte les contraintes sur le vecteur et la charge utile.

Par la suite, nous considérons un drone HALE qui a une caméra pour charge utile. Le cas des drones MALE est similaire.

Nous supposons connu l'état de la cible $q_{\text{cible}} = (x_{\text{cible}}, y_{\text{cible}})$, ainsi que ses dérivées $\dot{q}_{\text{cible}}$ et $\ddot{q}_{\text{cible}}$. Ces informations sont données par la cible elle-même (dans le cas du suivi de convoi ami), ou bien par un module d'estimation, cf. paragraphe 2.4.

L'idée est de considérer le problème comme une minimisation de l'erreur entre la vitesse courante de l'UAV $\dot{q}$ et la vitesse actuellement connue de la cible $\dot{q}_{\text{cible}}$ (e.g., un convoi à suivre).

Nous utilisons un modèle de Dubins légèrement modifié pour décrire la cinématique du drone HALE et la dynamique de sa caméra (dans le plan).

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = u \\ \dot{v} = \mu \\ \dot{\alpha} = u + \frac{1}{\tau_c}u_c - \frac{1}{\tau_c}(\alpha - \theta) \end{cases} \tag{3.1}$$

avec $q = (x, y, \theta, v, \alpha)$ l'état du système et $U = (u, \mu, u_c) \in \mathscr{U}$ ses commandes. Nous posons $\mathscr{U} = [-u_{\max}, u_{\max}] \times [-\mu_{\max}, \mu_{\max}] \times [\tilde{\alpha}_{\min}, \tilde{\alpha}_{\max}]$ avec $\mu_{\max}$, $u_{\max}$ positifs.

Remarquons ici que la vitesse du drone v peut évoluer grâce à la commande d'accélération μ. Nous la supposons contrainte et vérifiant $0 < v_{\min} < v < v_{\max}$.

Comme pour le système de Dubins classique, le point $(x, y) \in \mathbb{R}^2$ définit les coordonnées du drone dans le plan d'altitude constant et θ son cap. Les commandes μ et u sont associées au vecteur et régissent, respectivement, son accélération et sa commande de lacet. Un schéma est donné en Figure 3.1.

L'angle α régit le positionnement de la caméra dans le plan. Nous considérons que cet angle définit l'angle de la caméra dans le repère global, en d'autres termes $\alpha = \tilde{\alpha} + \theta$ où $\tilde{\alpha}$ est l'angle de la caméra dans le repère associé au drone. La commande u_c régit la position de la caméra dans ce repère, i.e. u_c commande $\tilde{\alpha}$. Nous supposons que la caméra est un système linéaire du premier ordre (la caméra est dirigée par un moteur électrique qui peut être considéré comme tel) :

$$\dot{\tilde{\alpha}} = \frac{K}{\tau_c}u_c - \frac{1}{\tau_c}\tilde{\alpha},$$

avec K le gain statique du système et τ_c sa constante de temps. Dans la suite, nous supposons que $K = 1$ ce qui peut toujours être effectué quitte à changer les bornes du contrôle u_c.

Remarque 3.1. L'équation de la caméra étant une équation différentielle du premier ordre, les contraintes de position sur $\tilde{\alpha}$ sont celles sur la commande u_c, i.e. nous supposons $\tilde{\alpha}_{\min} \leq \tilde{\alpha} \leq \tilde{\alpha}_{\max}$.

Grâce aux connaissances sur la cible, nous calculons α_{cible} et $\dot{\alpha}_{\text{cible}}$, où α_{cible} est l'angle que doit avoir la caméra pour observer la cible.

$$\alpha_{\text{cible}} = \arctan\left(\frac{y_{\text{cible}} - y}{x_{\text{cible}} - x}\right)$$
$$\dot{\alpha}_{\text{cible}} = \frac{(\dot{y}_{\text{cible}} - \dot{y})(x_{\text{cible}} - x) - (y_{\text{cible}} - y)(\dot{x}_{\text{cible}} - \dot{x})}{(x_{\text{cible}} - x)^2 + (y_{\text{cible}} - y)^2}$$

Remarque 3.2. Les cas particuliers où $x_{\text{cible}} - x = 0$ ou $(x_{\text{cible}} - x)^2 + (y_{\text{cible}} - y)^2 = 0$ sont traités à part. Dans le premier cas nous posons $\alpha_{\text{cible}} = \pi$. Lorsque $(x_{\text{cible}} - x)^2 + (y_{\text{cible}} - y)^2 = 0$ nous considérons que $\dot{\alpha}_{\text{cible}} = 0$.

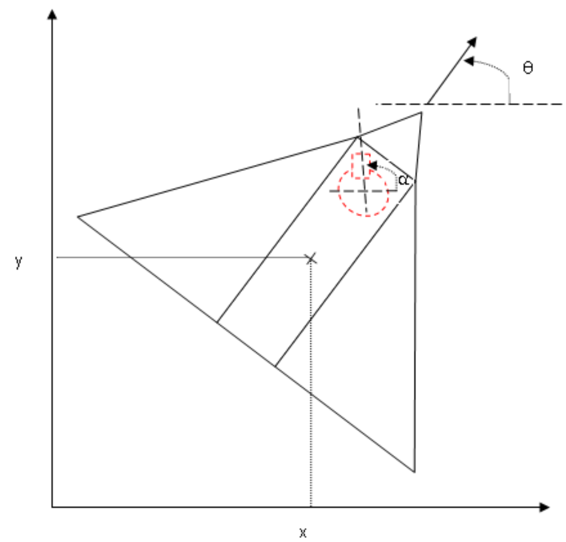


FIGURE 3.1 – Schéma du drone HALE et de la charge utile considérée

Nous minimisons les erreurs de positionnement ε et ε_c du vecteur et de la caméra, par rapport aux objectifs voulus. Ces erreurs sont définies de la manière suivante :

$$\varepsilon = \begin{bmatrix} x_{\text{cible}} - x \\ y_{\text{cible}} - y \end{bmatrix}$$
$$\varepsilon_c = \alpha_{\text{cible}} - \alpha$$

Le système régissant la position du vecteur (3.1) est invariant sous l'action du groupe des déplacements du plan, $SE(2)$ [8]. Nous supposons donc, en utilisant la méthode décrite par P. Rouchon [113], que l'erreur dynamique de suivi de cible peut s'exprimer comme un système linéaire d'ordre deux, i.e. l'erreur ε vérifie

$$\ddot{\varepsilon} = \iota_1 \dot{\varepsilon} + \iota_2 \varepsilon,$$

avec ι_1 et ι_2 deux constantes réelles.

En ce qui concerne l'erreur de suivi de la cible par la caméra, sachant que celle-ci est représentée par un système linéaire d'ordre 1, nous considérons que l'erreur est solution d'une équation différentielle linéaire d'ordre 1, i.e. l'erreur ε_c vérifie

$$\dot{\varepsilon}_c = \iota_3 \varepsilon_c,$$

avec ι_3 une constante réelle.

Les deux conditions sur l'évolution des erreurs conduisent au système suivant

$$\begin{cases} 0 = \ddot{\varepsilon} - \iota_1 \dot{\varepsilon} - \iota_2 \varepsilon \\ 0 = \dot{\varepsilon}_c - \iota_3 \varepsilon_c \end{cases}$$

La solution stationnaire $(\varepsilon, \varepsilon_c) = (0,0)$ est stable si $\iota_3 < 0$ et les valeurs propres de la matrice

$$A = \begin{bmatrix} 0 & 1 \\ \iota_2 & \iota_1 \end{bmatrix}$$

sont à parties réelles négatives.

De plus, en utilisant (3.1), nous avons

$$\begin{cases} \varepsilon_c = \alpha_{\text{cible}} - \alpha \\ \dot{\varepsilon}_c = \dot{\alpha}_{\text{cible}} - u - \dfrac{1}{\tau_c} u_c + \dfrac{1}{\tau_c}(\alpha - \theta) \end{cases}$$

et

$$\begin{cases} \varepsilon = \begin{pmatrix} x_{\text{cible}} \\ y_{\text{cible}} \end{pmatrix} - \begin{pmatrix} x \\ y \end{pmatrix} \\ \dot{\varepsilon} = \begin{pmatrix} \dot{x}_{\text{cible}} \\ \dot{y}_{\text{cible}} \end{pmatrix} - \begin{pmatrix} v\cos\theta \\ v\sin\theta \end{pmatrix} \\ \ddot{\varepsilon} = \begin{pmatrix} \ddot{x}_{\text{cible}} \\ \ddot{y}_{\text{cible}} \end{pmatrix} - \begin{pmatrix} -v\sin\theta \\ v\cos\theta \end{pmatrix} u - \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \mu. \end{cases}$$

En résumé, les commandes u, μ et u_c sont solutions du système de contrôle

$$\begin{aligned} 0 = &\begin{pmatrix} \ddot{x}_{\text{cible}} \\ \ddot{y}_{\text{cible}} \\ \dot{\alpha}_{\text{cible}} + \frac{1}{\tau_c}(\alpha - \theta) \end{pmatrix} - \iota_1 \begin{pmatrix} \dot{x}_{\text{cible}} - v\cos\theta \\ \dot{y}_{\text{cible}} - v\sin\theta \\ 0 \end{pmatrix} - \iota_2 \begin{pmatrix} x_{\text{cible}} - x \\ y_{\text{cible}} - y \\ 0 \end{pmatrix} - \iota_3 \begin{pmatrix} 0 \\ 0 \\ \alpha_{\text{cible}} - \alpha \end{pmatrix} \\ &- \begin{pmatrix} -v\sin\theta \\ v\cos\theta \\ 1 \end{pmatrix} u - \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} \mu - \begin{pmatrix} 0 \\ 0 \\ \frac{1}{\tau_c} \end{pmatrix} u_c, \end{aligned}$$

qui se réécrit sous la forme

$$0 = Z - RU, \tag{3.2}$$

où

$$\begin{aligned} Z &= \begin{pmatrix} \ddot{x}_{\text{cible}} \\ \ddot{y}_{\text{cible}} \\ \dot{\alpha}_{\text{cible}} + \frac{1}{\tau_c}(\alpha - \theta) \end{pmatrix} - \iota_1 \begin{pmatrix} \dot{x}_{\text{cible}} - v\cos\theta \\ \dot{y}_{\text{cible}} - v\sin\theta \\ 0 \end{pmatrix} - \iota_2 \begin{pmatrix} x_{\text{cible}} - x \\ y_{\text{cible}} - y \\ 0 \end{pmatrix} - \iota_3 \begin{pmatrix} 0 \\ 0 \\ \alpha_{\text{cible}} - \alpha \end{pmatrix}, \\ R &= \begin{bmatrix} -v\sin\theta & \cos\theta & 0 \\ v\cos\theta & \sin\theta & 0 \\ 1 & 0 & \frac{1}{\tau_c} \end{bmatrix}, \\ U &= (u, \mu, u_c). \end{aligned}$$

La commande $U^*(\cdot)$ solution du système (3.2) vérifie aussi le problème de minimisation quadratique suivant, pour tout $t \geq 0$

$$\left(Z(t) - R(t)U^*(t)\right)^T \Delta \left(Z(t) - R(t)U^*(t)\right) = \min_{U \in \mathscr{U}} \left[\left(Z(t) - R(t)U\right)^T \Delta \left(Z(t) - R(t)U\right)\right],$$

où Δ est une matrice de pondération que nous utilisons pour pondérer l'erreur de positionnement du vecteur par rapport à celle de la caméra, ou inversement.

Ce problème de minimisation a une solution explicite triviale si U n'est pas contraint. Dans le cas contraire, nous sommes dans le cadre de la programmation quadratique (e.g., la méthode simple de relaxation peut être appliquée [105]).

Afin d'asservir la vitesse v, que nous ne commandons pas directement, nous modifions les bornes de la commande d'accélération μ en fonction de la vitesse en cours. Ces deux bornes définissent donc deux fonctions à un paramètre telles que

$$\begin{aligned} \mu_- &: v \mapsto -\mu_{\max} \tanh(\sigma(v + v_{\min})) \\ \mu_+ &: v \mapsto \quad \mu_{\max} \tanh(\sigma(v - v_{\max})), \end{aligned}$$

avec σ une constante réelle.

3.1.2 Application de la stratégie de couplage vecteur/capteur

La méthode quadratique de recherche de trajectoires a été implantée sous Matlab. Les Figures 3.2, 3.3, 3.4, 3.5 sont des exemples de résultats de suivi de cibles. Elles correspondent à des contraintes différentes sur la caméra, ainsi qu'à des paramètres de couplage vecteur/capteur différents. C'est à dire que la matrice de pondération Δ est différente et le champ de vision de la caméra est contraint ou non.

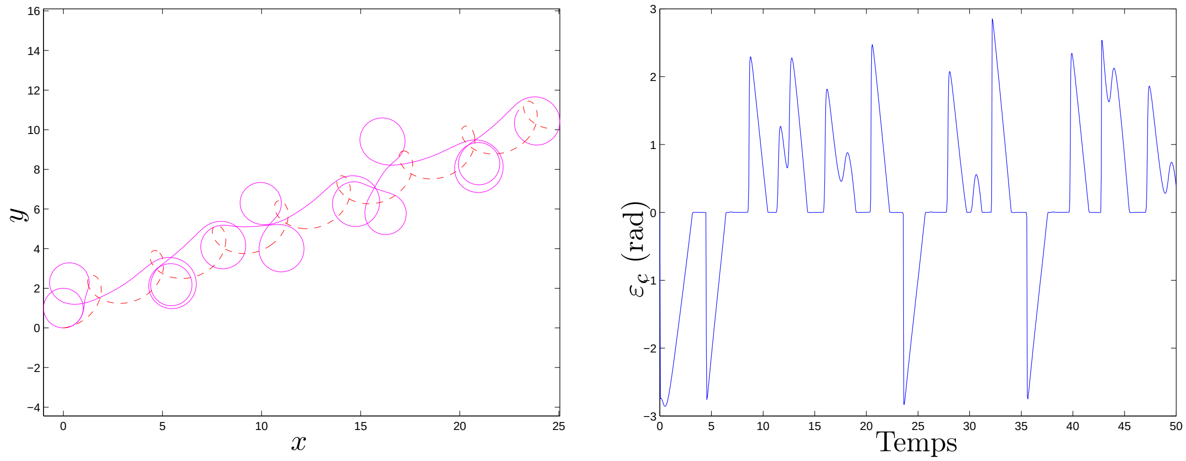
Sur chacune des Figures 3.2 à 3.6, la sous figure de gauche représente la trajectoire du vecteur obtenue par la méthode quadratique et la sous figure de droite représente l'erreur de positionnement de la caméra en fonction du temps. Sur chacune des sous figures de gauche, la trajectoire en trait continue est celle du vecteur et la trajectoire en traits pointillés correspond au mouvement de la cible à suivre. Les moyennes et écart-type des erreurs de positionnement de la caméra ($\bar{\varepsilon_c}$ et σ_{ε_c}, respectivement) et du vecteur ($\bar{\|\varepsilon\|}$ et $\sigma_{\|\varepsilon\|}$, respectivement) sont indiqués pour chaque simulation.

Les Figures 3.2 et 3.4 correspondent à des simulations avec une matrice de pondération qui donne une importance plus grande au positionnement de la caméra. À l'inverse, pour obtenir les Figures 3.3 et 3.5, la position du vecteur est jugée plus importante.

Les deux premières figures (les Figures 3.2 et 3.3) sont des simulations de planifications avec des contraintes sur le positionnement de la caméra. Ces contraintes n'ont pas été appliquées pendant les simulations correspondant aux Figures 3.4 et 3.5.

Un dernier exemple d'application est présenté sur la Figure 3.6. Lors de cette simulation, nous avons utilisé cette méthode pour la mission suivante :

- Le vecteur ne doit pas franchir la frontière (matérialisée par une ligne en tiret-point).
- La caméra doit suivre le pattern représenté par des tirets. L'angle d'observation de la caméra est supposé contraint : $\tilde{\alpha} \in [-\pi/3, \pi/3]$.



(a) $\|\bar{\varepsilon}\| = 2.0211$, $\sigma_{\|\varepsilon\|} = 1.2077$ (b) $\bar{\varepsilon_c} = 0.2226$, $\sigma_{\varepsilon_c} = 1.1529$

FIGURE 3.2 – Trajectoire primant l'erreur de positionnement de la caméra. Le champ de vision de la caméra est supposé contraint : $\tilde{\alpha} \in [-\pi/3, \pi/3]$.

Au regard des spécificités des caméras embarquées sur les drones [56], nous constatons que le positionnement de celles-ci a peu d'influence sur la trajectoire du vecteur.

Dans les simulations correspondant aux Figures 3.4 et 3.5, nous avons choisi des caméras de ce type.

Remarque 3.3. Dans la suite du travail, nous considérons uniquement le problème de planification de trajectoires du vecteur. En effet, comme la caméra embarquée peut suivre la cible sans difficulté, nous ne traitons que la planification du vecteur car cela est suffisant pour éviter que la cible ne sorte du champ de vision de la caméra (e.g., à cause de la limite de zoom, de l'occlusion de la cible...).

3.1.3 Découplage de la caméra

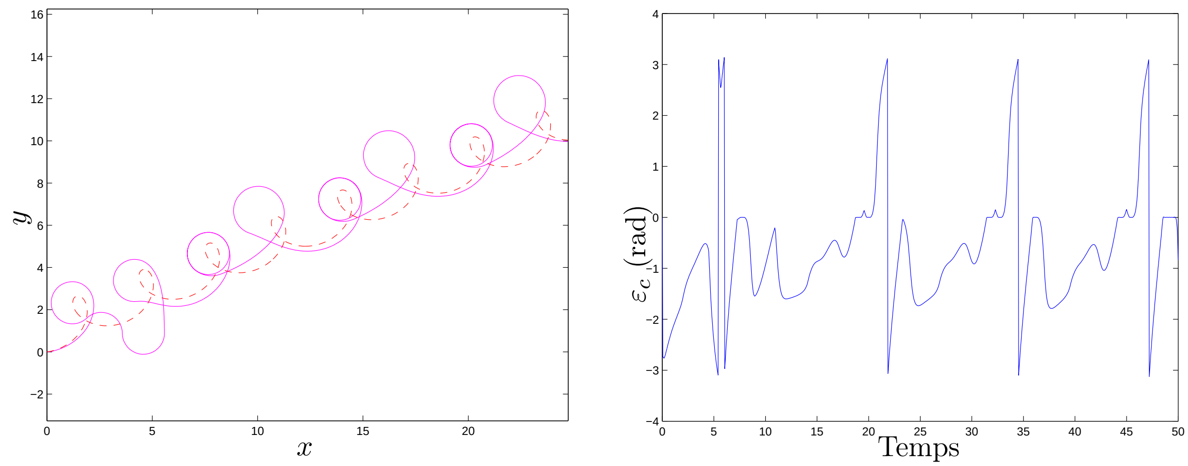
Dans le paragraphe précédent, nous avons conclu que la problématique du couplage vecteur/capteur se résume au guidage du vecteur (les caméras utilisées étant très performantes et non contraintes). C'est pourquoi, dans cette partie, nous proposons une méthode de commande indépendante de la caméra, que le problème soit 2D ou 3D.

3.1.3.1 Étude du cas plan

Nous modélisons le vecteur par le système de Dubins (2.2) et, comme au paragraphe 3.1.1, nous supposons que la caméra agit comme un système linéaire du premier ordre et qu'elle est commandée par une commande u_c, i.e.

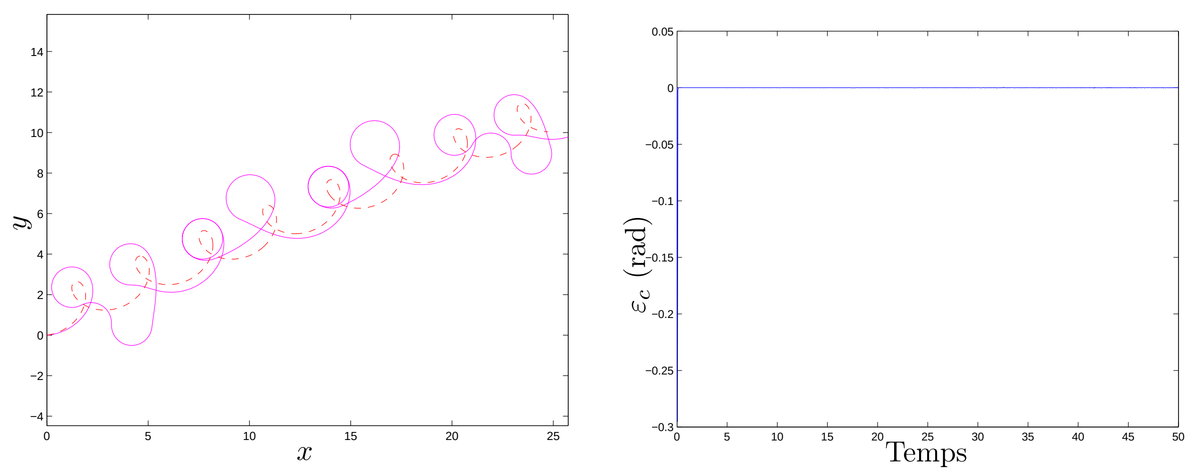
$$\dot{\alpha} = u + \frac{1}{\tau_c} u_c - \frac{1}{\tau_c}(\alpha - \theta)$$

avec α définissant l'angle de la caméra dans le repère global, i.e. $\alpha = \tilde{\alpha} + \theta$ où $\tilde{\alpha}$ est l'angle de la caméra dans le repère associé au drone. La caméra est supposée non contrainte, i.e. $\alpha \in \mathbb{R}$.



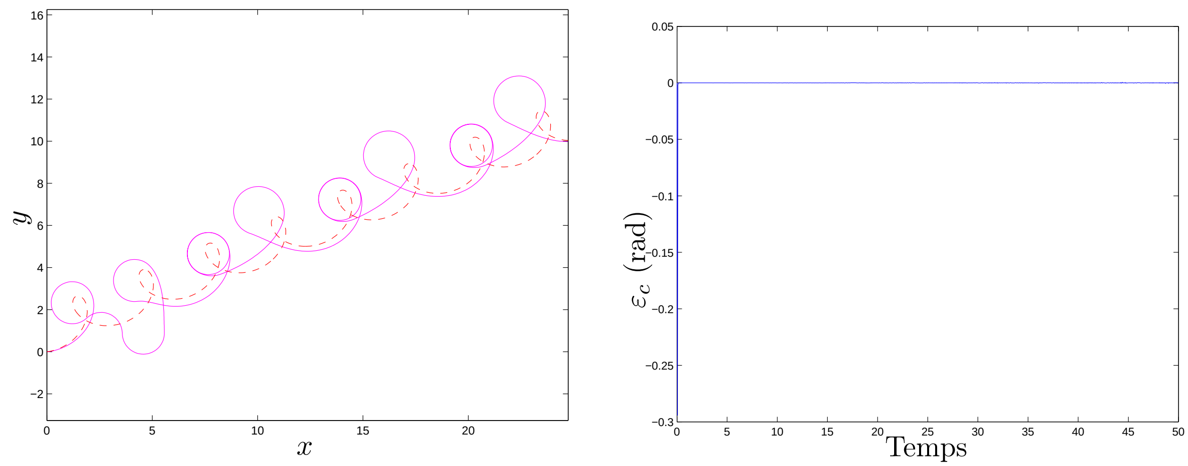
(a) $\|\bar{\varepsilon}\| = 1.3091$, $\sigma_{\|\varepsilon\|} = 0.6500$ (b) $\bar{\varepsilon_c} = -0.6579$, $\sigma_{\varepsilon_c} = 1.1576$

FIGURE 3.3 – Trajectoire primant l'erreur de positionnement du vecteur. Le champ de vision de la caméra est supposé contraint : $\tilde{\alpha} \in [-\pi/3, \pi/3]$.



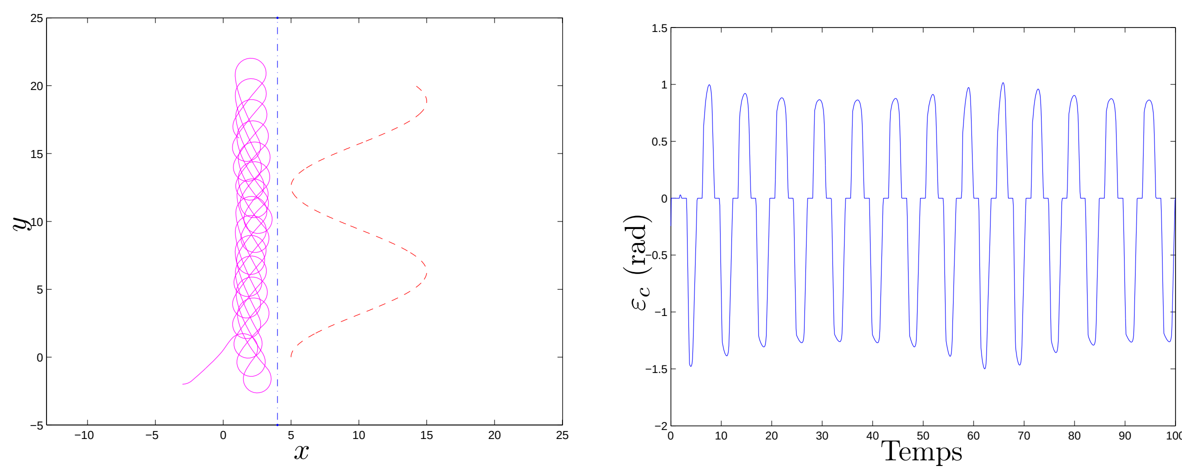
(a) $\|\bar{\varepsilon}\| = 1.2232$, $\sigma_{\|\varepsilon\|} = 0.6040$ (b) $\bar{\varepsilon_c} = -2.9664e{-4}$, $\sigma_{\varepsilon_c} = 0.0093$

FIGURE 3.4 – Trajectoire primant l'erreur de positionnement de la caméra. Le champ de vision de la caméra n'est pas contraint : $\tilde{\alpha} \in \mathbb{R}$.



(a) $\|\bar{\varepsilon}\| = 1.3092$, $\sigma_{\|\varepsilon\|} = 0.6507$ (b) $\bar{\varepsilon_c} = -2.9495e{-4}$, $\sigma_{\varepsilon_c} = 0.0093$

FIGURE 3.5 – Trajectoire primant l'erreur de positionnement du vecteur. Le champ de vision de la caméra n'est pas contraint.



(a) (b) $\bar{\varepsilon_c} = -0.1263$, $\sigma_{\varepsilon_c} = 0.7688$

FIGURE 3.6 – Sur cette figure les mouvements de la caméra sont contraints : $\tilde{\alpha} \in [-\pi/3, \pi/3]$. L'objectif est que la caméra suive le pattern en trait pointillés tout en assurant que la trajectoire ne franchisse pas la droite verticale.

Nous supposons connues la position et la vitesse de la cible, respectivement q_{cible} et $\dot{q}_{\text{cible}}$. Comme défini précédemment, nous souhaitons asservir l'angle de la caméra sur celui nous permettant de suivre (visuellement) la cible choisie. Notons α_{cible} cet angle, nous avons :

$$\begin{aligned}\alpha_{\text{cible}} &= \arctan\left(\frac{y_{\text{cible}} - y}{x_{\text{cible}} - x}\right)\\ \dot{\alpha}_{\text{cible}} &= \frac{(\dot{y}_{\text{cible}} - \dot{y})(x_{\text{cible}} - x) - (y_{\text{cible}} - y)(\dot{x}_{\text{cible}} - \dot{x})}{(x_{\text{cible}} - x)^2 + (y_{\text{cible}} - y)^2}\end{aligned}$$

Remarque 3.4. Si $x_{\text{cible}} - x = 0$ alors nous posons $\alpha_{\text{cible}} = \pi$. Si $(x_{\text{cible}} - x)^2 + (y_{\text{cible}} - y)^2 = 0$, nous considérons $\dot{\alpha}_{\text{cible}} = 0$.

Étant donné que l'équation de la caméra est un système linéaire d'ordre 1, nous considérons que l'erreur décrit un système différentiel d'ordre 1, i.e. l'erreur $\varepsilon_c = \alpha_{\text{cible}} - \alpha$ vérifie

$$\dot{\varepsilon}_c = \iota \varepsilon_c,$$

avec $\iota < 0$ une constante réelle qui nous permet d'obtenir la stabilité de la solution stationnaire $\varepsilon_c = 0$.

À cet effet, nous proposons la commande suivante [102]

$$u_c = \tau_c \left(\dot{\alpha}_{\text{cible}} - u + \frac{1}{\tau_c}(\alpha - \theta) - \iota \varepsilon_c\right).$$

3.1.3.2 Extension au cas 3D

Le cas d'un système évoluant dans l'espace à trois dimensions est légèrement différent, mais le principe reste le même.

Considérons le système d'équation (2.6), modélisant le drone MALE, comme présenté au paragraphe 2.1.2.

Nous supposons que la caméra a deux degrés de liberté en rotation qui nous permettent de modifier l'azimut θ_c (la rotation autour de son axe vertical, parallèle à l'axe $\vec{k}$) et l'angle d'élévation de la direction d'observation ψ_c, voir Figure 3.7.

Comme au paragraphe 3.1.1, nous supposons que la caméra est modélisé par deux systèmes linéaires du premier ordre découplés (de constantes de temps τ_{θ_c} et τ_{ψ_c} et de gains unitaires) et commandés par u_{θ_c} et u_{ψ_c}, i.e.

$$\begin{aligned}\dot{\theta}_c &= \frac{1}{\tau_{\theta_c}} u_{\theta_c} - \frac{1}{\tau_{\theta_c}} \theta_c\\ \dot{\psi}_c &= \frac{1}{\tau_{\psi_c}} u_{\psi_c} - \frac{1}{\tau_{\psi_c}} \psi_c,\end{aligned}$$

Nous souhaitons asservir la direction d'observation de la caméra à celle nous permettant de suivre (visuellement) la cible choisie. La vitesse et la position de cette dernière sont supposées

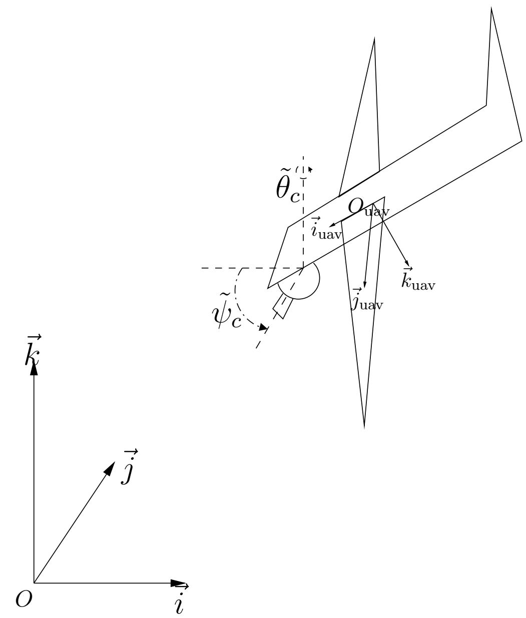


FIGURE 3.7 – Schéma du drone et de sa caméra en 3D

connues. Cette direction a une composante d'azimut θ_{cible} et une d'élévation ψ_{cible}. Nous obtenons ces composantes ainsi que leurs dérivées de la manière suivante,

$$\theta_{\text{cible}} = \arctan\left(\frac{\Delta x_2}{\Delta x_1}\right)$$

$$\dot{\theta}_{\text{cible}} = \frac{\dot{\widehat{\Delta x_2}}\Delta x_1 - \Delta x_2 \dot{\widehat{\Delta x_1}}}{(\Delta x_1)^2 + (\Delta x_2)^2}$$

et

$$\psi_{\text{cible}} = \arctan\left(\frac{\Delta x_3}{\sqrt{(\Delta x_1)^2 + (\Delta x_2)^2}}\right)$$

$$\dot{\psi}_{\text{cible}} = \frac{\left((\Delta x_1)^2 + (\Delta x_2)^2\right)\dot{\widehat{\Delta x_3}} - \Delta x_3\left(\Delta x_1 \dot{\widehat{\Delta x_1}} + \Delta x_2 \dot{\widehat{\Delta x_2}}\right)}{\sqrt{(\Delta x_1)^2 + (\Delta x_2)^2}\left((\Delta x_1)^2 + (\Delta x_2)^2 + (\Delta x_3)^2\right)}$$

avec $\Delta x_i = x_{i_{\text{cible}}} - x_i$ et x_i, $x_{i_{\text{cible}}}$, $i = \{1, 2, 3\}$ les coordonnées, respectivement, de la caméra et de la cible dans le repère global.

Remarque 3.5. Si $\Delta x_1 = 0$ alors nous posons $\theta_{\text{cible}} = \pi$. Si $(\Delta x_1)^2 + (\Delta x_2)^2 = 0$, nous considérons $\psi_{\text{cible}} = \pi/2$, $\dot{\psi}_{\text{cible}} = 0$ et $\dot{\theta}_{\text{cible}} = 0$.

La caméra étant un système linéaire d'ordre 1, nous considérons que l'erreur décrit aussi un système différentiel d'ordre 1, i.e. les erreurs $\varepsilon_{\theta_c} = \theta_{\text{cible}} - \theta_c$ et $\varepsilon_{\psi_c} = \psi_{\text{cible}} - \psi_c$ vérifient

$$\dot{\varepsilon}_{\theta_c} = \iota \varepsilon_{\theta_c}$$
$$\dot{\varepsilon}_{\psi_c} = \iota \varepsilon_{\psi_c}$$

avec $\iota < 0$ une constante réelle qui nous permet d'avoir la stabilité des solutions désirées.

Ces objectifs sont réalisés en utilisant les commandes suivantes [102]

$$u_{\theta_c} = \tau_{\theta_c} \left(\dot{\theta}_{\text{cible}} + \frac{1}{\tau_{\theta_c}} \theta_c - \iota \varepsilon_{\theta_c} \right)$$
$$u_{\psi_c} = \tau_{\psi_c} \left(\dot{\psi}_{\text{cible}} + \frac{1}{\tau_{\psi_c}} \psi_c - \iota \varepsilon_{\psi_c} \right)$$

3.2 Contrôle de type Lyapunov et temps-minimal pour les drones

L'article présenté dans ce paragraphe est en cours de publication.

Lyapunov and minimum-time path planning for drones

Thibault Maillot* Ugo Boscain†‡ Jean-Paul Gauthier*‡ Ulysse Serres§

July 18, 2013

Abstract

In this paper, we study the problem of controlling an unmanned aerial vehicle (UAV) to provide a target supervision and/or to provide convoy protection to ground vehicles.

We first present a control strategy based upon a Lyapunov-LaSalle stabilization method to provide supervision of a stationary target. The UAV is expected to join a pre-designed admissible circular trajectory around the target which is itself a fixed point in the space.

Our strategy is presented for both HALE (High Altitude Long Endurance) and MALE (Medium Altitude Long Endurance) types UAVs. A UAV flying at a constant altitude (HALE type) is modeled as a Dubins vehicle (i.e. a planar vehicle with constrained turning radius and constant forward velocity). For a UAV that might change its altitude (MALE type), we use the general kinematic model of a rigid body evolving in $\mathbb{R}^3$. Both control strategies presented are smooth and unlike what is usually proposed in the literature these strategies asymptotically track a circular trajectory of exact minimum turning radius.

We also present the time-optimal control synthesis for tracking a circle by a Dubins vehicle. This optimal strategy, although much simpler than the point-to-point time-optimal strategy obtained by P. Souères and J.-P. Laumond in the 1990s (see [45]), is very rich.

Finally, we propose control strategies to provide supervision of a moving target, that are based upon the previous ones.

Keywords: optimal control, path planning, aircraft navigation, unmanned aerial vehicles, rigid-body dynamics, under-actuated systems, nonlinear control, trajectory tracking.

1 Introduction

This study holds in the context of the French FUI SHARE project (see [3]), supported by a consortium of companies and research labs [1], and the authors are granted from the project. The aim is to design a trajectory for a UAV from some starting point and orientation to some final submanifold of the state space.

The UAV path planning problem is a standard motion planning problem and although the bibliography on this subject is very rich we do not aim to provide an exhaustive list. Let us just mention that classical planning algorithms are based upon very different approaches such as geometric control [14, 19, 32], optimal control [21], flatness [7], stochastic theory [5].

*Université du Sud-Toulon-Var, LSIS, UMR CNRS 7296, B.P 20132, 83957 La Garde Cedex, France; email: thibault.maillot@univ-tln.fr & jean-paul.gauthier@univ-tln.fr

†CNRS, Centre de Mathématiques Appliquées - Ecole Polytechnique (CMAP), UMR CNRS 7641, Route de Saclay, 91128 Palaiseau Cedex, France; email: ugo.boscain@polytechnique.edu

‡INRIA GECO Project

§Université de Lyon, F-69 622 Lyon, France; Université Lyon 1, Villeurbanne; LAGEP, UMR CNRS 5007, 43 bd du 11 novembre 1918, 69100 Villeurbanne, France; email: ulysse.serres@univ-lyon1.fr

[1] Opéra Ergonomie, ONERA, Thales Alénia Space, Eurocopter, Adetel group.

Moreover, the reader may refer to [27] for a survey on trajectory-planning algorithms focusing on autonomous UAV guidance.

The purpose of this paper is to present different solutions to the above mentioned motion planning problem for both HALE and MALE types of fixed-wing UAVs. Although the case of rotary-wing drones is interesting, it is simpler and more standard motion planning methods may be applied (see [8, 26]).

The problem is addressed from a kinematic point of view only. In particular, we consider that the aircraft controls are just the angular velocities. From a kinematic point of view, a UAV can be specified by a control system $\dot{q} = f(q, u)$ (specific models will be given later on), where q denotes the state of the UAV which usually includes the UAV position (in $\mathbb{R}^2$ or $\mathbb{R}^3$) and attitude (i.e. the UAV orientation in S^1 or $SO(3)$), and u is the control vector driving the UAV kinematics, such as angular and spacial velocities. Moreover, we make the following assumptions on the UAV:

- the velocity of the UAV is assumed to have a positive lower bound (no stationary or quasistationary flights are allowed) and a positive upper bound;
- the UAV is assumed to be kinematically restricted by its minimum turning radius $r > 0$, or equivalently, its yaw angle is assumed to be constrained by an upper positive bound.

In this article, tools from Lyapunov-based stability analysis and optimal control theory are used for the development of control algorithms achieving the desired motion planning task.

The paper is organized as follows: in Section 2, we present global Lyapunov-LaSalle-based stabilization results for a UAV tracking a horizontal circle of minimal turning radius. Section 2.1 and Section 2.2 are the detailed studies of the cases of HALE and MALE UAVs respectively.

Section 3 describes the time-optimal synthesis for the Dubins system tracking a minimal radius circle. The study is done in a reduced state space. On this subject, we have to mention the great work [45] about the more complicated point-to-point problem (see also [1] for a partial study).

Finally, in Section 4 the results of Sections 2 and 2.2 are brought together and an algorithm for the tracking of a moving target is proposed. In the present paper, what is meant by *target* is the position (in $\mathbb{R}^2$ or $\mathbb{R}^3$) of the center of mass of the object the UAV has to follow.

We end this introductory section providing a list of mathematical conventions used here in. In the paper, smooth means C^∞. Throughout the remaining of the paper, the time dependence of variables is often omitted to lighten notations. Moreover, we use

- the dot $\dot{}$ to denote the derivative with respect to time;
- $|x|$ to denote the usual Euclidean norm of $x \in \mathbb{R}^n$, and $\langle x, y \rangle$ for the inner product of two such vectors;
- $B(x, r) \subset \mathbb{R}^n$ to denote the open ball of radius r centered at x;
- M' to denote the transpose of a matrix M;
- $\mathrm{tr}(M) = \sum_{i=1}^n M_{ii}$ to denote the trace of a $n \times n$ square matrix M;
- $[\cdot, \cdot]$ to denote the Lie bracket between vector fields and the commutator of matrices.

- For two $n \times n$ real matrices M and N, the Hilbert-Schmidt scalar product is $\mathrm{tr}(M'N)$, then the Hilbert-Schmidt norm of a matrix M is $\sqrt{\mathrm{tr}(M'M)}$.
- Also, we do not distinguish between row and column vectors, the distinction being clear from the context.

2 Lyapunov-LaSalle-based stabilization

This section is concerned with the problem of global stabilization of a UAV (in the position-attitude state space) on a fixed horizontal circle of minimum turning radius with prescribed direction that we shall refer to as the *final manifold*.

By using a standard Lyapunov-LaSalle-based approach, we propose, for both HALE and MALE type UAVs, new nonlinear feedback control laws that stabilize the UAV on the final manifold.

The Lyapunov method has been used by several authors, in different ways such as for example [15, 16, 20, 22, 24, 28, 33, 35, 50]. Here we present an original Lyapunov strategy, serving the advantage to be completely smooth, very simple to apply, and with the peculiarity that the circular final manifold is with exactly minimum turning radius. Frequently, Lyapunov strategies presented in the literature do not have this last feature.

Along this section, controls are assumed to be smooth. Nonsmooth controls appear in Section 3 only.

2.1 Lyapunov based method for the Dubins system

2.1.1 Problem under consideration

From the kinematic point of view, a rough HALE drone is governed by the standard Dubins equations (see e.g. [18] for a justification, for Dubins problems on Riemannian manifolds see [17, 43]):

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{\theta} = u. \end{cases} \tag{2.1}$$

with $(x, y, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1$ being the state (where $(x, y) \in \mathbb{R}^2$ is the UAV's coordinate in the constant altitude plane, and θ the yaw angle), and $u \in [-u_{\max}, u_{\max}]$ being the control variable. Note that the yaw angle θ is the angle made by the aircraft direction with respect to the x-axis.

These equations express that the drone evolves on a perfect plane (perfect constant altitude), at perfect constant speed 1, moves in the direction of its velocity vector, and is able to turn right and left.

Note that the rough model (2.1) is pertinent for HALE drones, the velocity being almost really constant.

As said in the introduction, the UAV is assumed to be kinematically restricted by its positive minimum turning radius r which implies the following bound on the yaw angular velocity:

$$-\frac{1}{r} = -u_{\max} \leqslant u \leqslant u_{\max} = \frac{1}{r}.$$

Remark 2.1. Note that up to a dilation in the (x, y)-plane we may assume without loss of generality that $[-u_{\max}, u_{\max}] = [-1, 1]$. This normalization could be used to simplify the treatment. We do this in Section 3.

We denote by $\mathcal{C}$ the final manifold which is defined to be the counterclockwise oriented circle centered at the origin of the (x, y)-plane, with radius $r = 1/u_{\max}$. In the (x, y, θ)-coordinates system, $\mathcal{C}$ is given by

$$\mathcal{C} = \{(x, y, \theta) \mid x = r\sin\theta,\ y = -r\cos\theta\}.$$

We have the following result.

Theorem 2.2. *There exists a smooth feedback control $k : \mathbb{R}^2 \times \mathbb{S}^1 \to [-u_{\max}, u_{\max}]$ such that the set $\mathcal{C}$ is a globally asymptotically stable attractor for the closed-loop system resulting from applying the feedback control $u = k(x, y, \theta)$ to system* (2.1).

2.1.2 Proof of theorem 2.2

The proof of Theorem 2.2 is based upon a standard Lyapunov-based control design and LaSalle's principle (see e.g. [25, Theorem 3.5, p.190] or the original LaSalle's paper [31]).

To construct a state feedback stabilizer to $\mathcal{C}$ for system (2.1) (and further to construct the time-optimal synthesis in Section 3), it is convenient to use UAV-based coordinates $(\tilde{x}, \tilde{y}, \theta)$ with $\tilde{x}$ and $\tilde{y}$ defined by

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix},$$

in which the set $\mathcal{C}$ takes the form

$$\tilde{\mathcal{C}} = \{(\tilde{x}, \tilde{y}, \theta) \mid \tilde{x} = 0,\ \tilde{y} = -r\}.$$

Notice that if $\mathcal{C}$ would have been travelled clockwise, the equation $\tilde{y} = -r$ would have been changed for $\tilde{y} = r$.

In the $(\tilde{x}, \tilde{y}, \theta)$-coordinates system, the variable θ is decoupled and the two first equations of system (2.1) may be rewritten as

$$\begin{cases} \dot{\tilde{x}} = u\tilde{y} + 1 \\ \dot{\tilde{y}} = -u\tilde{x}. \end{cases} \tag{2.2}$$

Remark 2.3. One can read, in (2.2) that if u is changed for $-u$, the two equilibria of (2.2) corresponding to the control values $-u_{\max}$ and $u_{\max}$ are exchanged so that the set of equilibria remains unchanged. It means that we can indifferently consider one of the two equilibria positions.

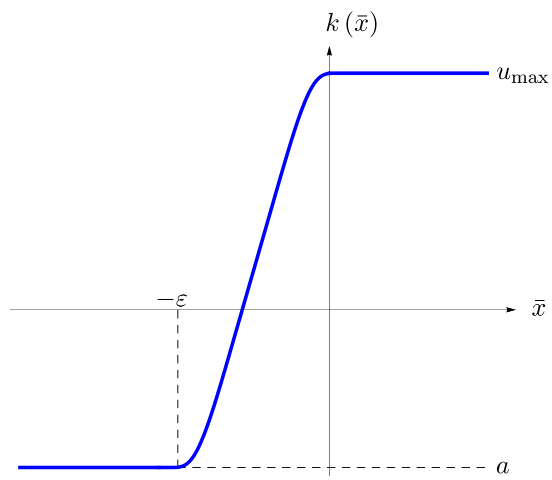
Changing the $(\tilde{x}, \tilde{y})$ coordinates for $(\bar{x}, \bar{y}) = (\tilde{x}, \tilde{y} + r)$, the stabilization problem to $\mathcal{C}$ is reformulated in these variables as the stabilization problem to the submanifold $\{(\bar{x}, \bar{y}, \theta) \mid \bar{x} = \bar{y} = 0\}$. Equivalently, we will refer to the convergence of $(\bar{x}, \bar{y})$ to the point $(0, 0)$ of the *reduced state space*. The reader can easily check that the $(\bar{x}, \bar{y})$ obey the following equations:

$$\begin{cases} \dot{\bar{x}} = u\bar{y} + 1 - ru \\ \dot{\bar{y}} = -u\bar{x}. \end{cases} \tag{2.3}$$

The proof of Theorem 2.2 is a consequence of its analogue in the reduced $(\bar{x}, \bar{y})$-coordinate state space.

Lemma 2.4. *There exists a (smooth) feedback control $k : \mathbb{R}^2 \to [-u_{\max}, u_{\max}]$ which stabilizes trajectories of the system* (2.3) *with $u(t) = \bar{k}(\bar{x}(t), \bar{y}(t))$ with respect to the origin.*

Proof. We shall prove that
$$V(\bar{x},\bar{y}) = \bar{x}^2 + \bar{y}^2$$
is a Lyapunov function for system (2.3) if the control law is well chosen. A straightforward computation yields
$$\dot{V}(\bar{x},\bar{y}) = 2\bar{x}\,(1 - ru)\,.$$
To be a Lyapunov function for system (2.3), we need to ensure that $\dot{V}$ is nonpositive. Keeping in mind that $u_{\max} = 1/r$, it yields $u = u_{\max}$ when $\bar{x} \geqslant 0$. Let ε be a positive real number, let a be such that $-u_{\max} \leqslant a < u_{\max}$ and let $k : \mathbb{R}^2 \to [-u_{\max}, u_{\max}]$ be a (actually any) smooth feedback control which satisfies
$$\bar{k}(\bar{x},\bar{y}) = \begin{cases} u_{\max} & \text{if } \bar{x} \geqslant 0 \\ -u_{\max} \leqslant \bar{k}(\bar{x},\bar{y}) < -u_{\max} & \text{if } \bar{x} < 0. \end{cases} \tag{2.4}$$
For instance, one can take the function defined by
$$\bar{k}(\bar{x},\bar{y}) = \begin{cases} a & \text{if } \bar{x} \leqslant -\varepsilon \\ \frac{u_{\max}-a}{1+e^{1/(\bar{x}+\varepsilon)+1/\bar{x}}} + a & \text{if } \bar{x} \in (-\varepsilon, 0) \\ u_{\max} & \text{if } \bar{x} \geqslant 0, \end{cases}$$
the shape of which is drawn on the next figure.



According to LaSalle's principle and since V is a proper function, we infer that all the trajectories of system (2.3) with feedback control $\bar{k}(\cdot)$ converge to the largest invariant set contained in the set $E = \{(\bar{x},\bar{y}) \mid \dot{V}(\bar{x},\bar{y}) = 0\} = \{(\bar{x},\bar{y}) \mid \bar{x} \geqslant 0\}$. Due to condition (2.4) on the feedback control, it is easy to see that any solution to (2.3) starting at $(\bar{x}_0, \bar{y}_0)$ with $\bar{x}_0 > 0$ escapes from E in finite time. Indeed, when $x_0 > 0$ the solution to the closed-loop system satisfies on E:
$$\begin{cases} \dot{\bar{x}} = u_{\max}\bar{y} \\ \dot{\bar{y}} = -u_{\max}\bar{x}. \end{cases}$$

Hence, the solution $(\bar{x}(t), \bar{y}(t)) = (\bar{x}_0 \cos(u_{\max}t) + \bar{y}_0 \sin(u_{\max}t), \bar{y}_0 \cos(u_{\max}t)) - \bar{x}_0 \sin(u_{\max}t)$ escapes from E at time
$$t = \begin{cases} \frac{-1}{u_{\max}} \arctan\left(\frac{\bar{x}_0}{\bar{y}_0}\right) & \text{if } \bar{y}_0 < 0 \\ \frac{\pi}{2u_{\max}} & \text{if } \bar{y}_0 = 0 \\ \frac{1}{u_{\max}}\left(\pi - \arctan\left(\frac{\bar{x}_0}{\bar{y}_0}\right)\right) & \text{if } \bar{y}_0 > 0. \end{cases}$$

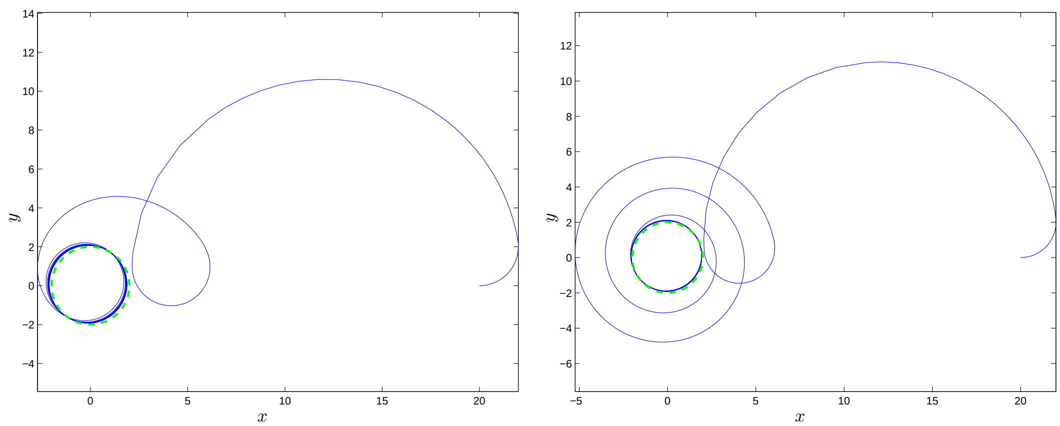
Consequently, the largest invariant set contained in E is the equilibrium point $(0, 0)$. This ends the proof. ■

Proof of Theorem 2.2. Since $V(\bar{x}, \bar{y})$ tends to zero implies that (x, y, θ) tends to $\mathcal{C}$, Lemma (2.4) shows that $\bar{k}$ steers asymptotically system (2.3) to $\mathcal{C}$, or equivalently that $k(x, y, \theta) = \bar{k}(\phi(x, y, \theta)$, (where $\phi : (x, y, \theta) \mapsto (\bar{x}, \bar{y})$) steers asymptotically system (2.1) to $\mathcal{C}$. ■

2.1.3 Numerical results

The theory exposed in the preceding subsections was tested by means of simulated data. During these simulations, we numerically integrate the closed-loop system resulting from applying the feedback control $u = k(x, y, \theta)$ of Theorem 2.2 to system (2.1) during 200 units of time.

Figure 1 shows two trajectories of the same UAV starting from the same initial point but with two different values of the parameter ε in the control function defined by equation (2.4). On both pictures the UAV reaches the final manifold tangentially, here the circle of radius 2 centered at the origin.



(a) (b)

Figure 1: Illustration of Lyapunov-LaSalle's-based 2D-planification results with starting point $(x_0, y_0, \theta_0) = (20, 0, 0)$, control constraints $u_{\max} = 0.5$ and $a = 0.1$. (a) $\varepsilon = 2$, (b) $\varepsilon = 0.5$.

2.2 Extension of the method to the 3D-case

2.2.1 Statement of the problem and of the result

In this section, we extend the results of the previous section to the case of MALE UAVs flying at constant speed. Our goal is unchanged: we want to (globally) stabilize the UAV on a final manifold.

From the kinematic point of view, a rough MALE drone (modeled by a standard 6 degree of freedom solid, symmetric w.r.t. its vertical middle plane) is governed by the following equations

$$\begin{cases} \dot{X} = RV_0 \\ \dot{R} = R\Omega \end{cases} \tag{2.5}$$

with

- $Q = (X, R) \in \mathbb{R}^3 \times SO(3)$ being the state where X is the UAV's coordinates expressed in the inertial frame and R, the UAV attitude, is the rotation matrix between the UAV's fixed frame and the inertial frame which gives the motion of the UAV's frame orientation;
- $V_0 = (1, 0, 0)$ being the thrust direction expressed in the UAV's frame;
- and $\Omega = \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix} \in \mathfrak{so}(3)$ (the Lie algebra of $SO(3)$ consisting of skew-symmetric matrices) the angular velocity of the body expressed in the body-fixed frame.

System (2.5) is used by many authors in the literature to model UAVs and airplanes (see e.g. [29, 48]). See also [49] and references therein. We define $u = (u_1, u_2, u_3)$ to be the control variables and we denote by $U = \prod_{i=1}^{3}[-u_{i\,\max}, u_{i\,\max}]$ the control set. The control variables are the roll (u_1), pitch (u_2) and yaw (u_3) angular velocities and are constrained by $u_{i\,\max} > 0$ for $i = 1, 2, 3$.

Remark 2.5. Notice that the controllability of system (2.5) with control in U is a direct consequence of [9, Theorem 1] (see also [40, Theorem 6.7]).

Equations (2.5) express that the drone moves in a 3-dimensional space, at perfect constant speed 1, moves in the direction of its velocity vector, and is able to turn in every direction of its fixed frame.

The First equation of system (2.5), in $\mathbb{R}^3$, describes the evolution of the UAV in space. The second one, in $SO(3)$, depicts the motion of the UAV's frame which is controlled by three controls on angular velocity acting physically on the roll, pitch and yaw angles.

Set $X = (x_1, x_2, x_3)$. The (x_1, x_2)-plane is called the horizontal plane. Here, we fix the final manifold $\mathcal{C}$ to be the union $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$ of the two circles of radius $r_3 = 1/u_{3\,\max}$ centered at the origin in the horizontal (x_1, x_2)-plane and oriented counterclockwise in the orientation given by their normals $N_1 = (0, 0, 1)$, $N_2 = (0, 0, -1)$. Define

$$J_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \quad J_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \quad J_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

so that $\Omega = u_1 J_1 + u_2 J_2 + u_3 J_3$.

Notice that the Lie algebra structure on $\mathfrak{so}(3)$ is given by

$$[J_1, J_2] = J_3, \quad [J_2, J_3] = J_1, \quad [J_3, J_1] = J_2. \tag{2.6}$$

The reader can easily check that, in the (X, R)-coordinates system, the final manifold takes the form: $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$, setting $\Delta_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$,

$$\mathcal{C}_1 = \{(X, R) \in \mathbb{R}^3 \times SO(3) \mid X = r_3(\sin\theta, -\cos\theta, 0),\ R = e^{J_3\theta},\ \theta \in \mathbb{R}\}$$

$$\mathcal{C}_2 = \{(X, R) \in \mathbb{R}^3 \times SO(3) \mid X = r_3(\sin\theta, \cos\theta, 0),\ R = \Delta_1 e^{J_3\theta},\ \theta \in \mathbb{R}\}.$$

We will prove the following result.

Theorem 2.6. *There exists a (smooth) feedback control $k : \mathbb{R}^3 \times SO(3) \to U$ such that the set $\mathcal{C}$ is a global asymptotic attractor for the closed-loop system resulting from applying the feedback control $u = k(X, R)$ to system (2.5). Moreover, $\mathcal{C}_1$ is an asymptotically stable limit cycle and $\mathcal{C}_2$ is an unstable limit cycle. All trajectories tend to $\mathcal{C}_1$ but the trajectories starting in the plane $x_3 = 0$ with $R(0) = \Delta_1 e^{J_3\theta}$ for some θ, which tend to $\mathcal{C}_2$.*

2.2.2 Proof of Theorem 2.6

Here again, to construct a state feedback stabilizer to $\mathcal{C}$ for system (2.5) it is convenient to use UAV-based coordinates $(\tilde{X}, R)$ defined by $X = R\tilde{X}$ in which $\tilde{X}$ obeys

$$\dot{\tilde{X}} = V_0 - \Omega\tilde{X},$$

and the final manifold takes the form $\tilde{\mathcal{C}} = \tilde{\mathcal{C}}_1 \cup \tilde{\mathcal{C}}_2$,

$$\tilde{\mathcal{C}}_1 = \left\{ \left(\tilde{X}, R\right) \in \mathbb{R}^3 \times SO(3) \mid \tilde{X} = (0, -r_3, 0),\ R = e^{J_3\theta},\ \theta \in \mathbb{R} \right\},$$

$$\tilde{\mathcal{C}}_2 = \left\{ \left(\tilde{X}, R\right) \in \mathbb{R}^3 \times SO(3) \mid \tilde{X} = (0, -r_3, 0),\ R = \Delta_1 e^{J_3\theta},\ \theta \in \mathbb{R} \right\}.$$

Set $\tilde{X}^* = (0, -r_3, 0)$ and consider the new variable

$$\bar{X} = \tilde{X} - \tilde{X}^*.$$

It yields the new system

$$\begin{cases} \dot{\bar{X}} = V_0 - \Omega\left(\bar{X} + \tilde{X}^*\right) \\ \dot{R} = R\Omega. \end{cases} \tag{2.7}$$

In particular, we have

$$\begin{cases} \dot{\bar{x}}_1 = u_3\bar{x}_2 - u_2\bar{x}_3 + (1 - r_3u_3) \\ \dot{\bar{x}}_2 = -u_3\bar{x}_1 + u_1\bar{x}_3 \\ \dot{\bar{x}}_3 = u_2\bar{x}_1 - u_1\bar{x}_2 + r_3u_1. \end{cases}$$

The stabilization problem to $\tilde{\mathcal{C}}$ is reformulated in these variables as the stabilization problem to the submanifold $\bar{\mathcal{C}} = \bar{\mathcal{C}}_1 \cup \bar{\mathcal{C}}_2$,

$$\bar{\mathcal{C}}_1 = \left\{ \left(\bar{X}, R\right) \in \mathbb{R}^3 \times SO(3) \mid \bar{X} = 0, \quad R = e^{J_3\theta}, \quad \theta \in \mathbb{R} \right\}.$$

$$\bar{\mathcal{C}}_2 = \left\{ \left(\bar{X}, R\right) \in \mathbb{R}^3 \times SO(3) \mid \bar{X} = 0, \quad R = \Delta_1 e^{J_3\theta}, \quad \theta \in \mathbb{R} \right\}.$$

Let σ be a positive real number. We shall prove that the function $V : \mathbb{R}^3 \times SO(3) \to \mathbb{R}_+$ defined by

$$\begin{aligned} V(\bar{X}, R) &= \frac{1}{2}\left(\sigma\bar{X}'\bar{X} + \operatorname{tr}\left([R, J_3]'[R, J_3]\right)\right) \\ &= V_1(\bar{X}, R) + V_2(\bar{X}, R), \end{aligned} \tag{2.8}$$

with $V_1(\bar{X}, R) = \frac{\sigma}{2}\bar{X}'\bar{X}$ and $V_2(\bar{X}, R) = \frac{1}{2}\operatorname{tr}\left([R, J_3]'[R, J_3]\right)$, is a Lyapunov function for the closed-loop system resulting from applying a well-chosen feedback control $u = k(\bar{X}, R)$ to (2.7).

Remark 2.7. Notice that the coefficient σ in the Lyapunov function (2.8) is a weight parameter that can be tuned in order to give more (resp. less) importance to the horizontality (i.e., pitch and roll angles close to zero) of the drone rather than its distance (in $\mathbb{R}^3$) to the target if $\sigma < 1$ (resp. $\sigma > 1$) during the stabilization process. Although this parameter plays no role in the subsequent results and computations, we keep it in our formulas.

The function V is clearly smooth. Moreover, the following proposition is obvious.

Proposition 2.8. *$V(\cdot)$ is a proper function.*

Notice that $V\left(\bar{X},R\right)$ may be rewritten as

$$V(\bar{X},R)=\frac{\sigma}{2}\bar{X}'\bar{X}+2+\operatorname{tr}\left(RJ_3R'J_3\right).$$

Hence, taking into account equations (2.6), the derivation w.r.t. time of $V\left(\bar{X},R\right)$ along the trajectories of system (2.7) yields

$$\begin{aligned}
\dot{V}\left(\bar{X},R\right) &= \sigma\bar{X}'\dot{\bar{X}}+\frac{d}{dt}\operatorname{tr}\left(RJ_3R'J_3\right)\\
&= \sigma\bar{X}'\big(V_0-\Omega\big(\bar{X}+\tilde{X}^*\big)\big)+\operatorname{tr}\left(R[\Omega,J_3]R'J_3\right)\\
&= \sigma\bar{X}'\big(V_0-\Omega\tilde{X}^*\big)+\operatorname{tr}\big(R(u_1[J_1,J_3]+u_2[J_2,J_3]R)J_3\big)\\
&= \sigma\left\langle\begin{pmatrix}\bar{x}_1\\\bar{x}_2\\\bar{x}_3\end{pmatrix},\begin{pmatrix}1\\0\\0\end{pmatrix}+(u_1J_1+u_3J_3)\begin{pmatrix}0\\r_3\\0\end{pmatrix}\right\rangle-u_1\operatorname{tr}\left(RJ_2R'J_3\right)+u_2\operatorname{tr}\left(RJ_1R'J_3\right)\\
&= \sigma\bar{x}_1\left(1-r_3u_3\right)+u_1\left(\sigma r_3\bar{x}_3-\operatorname{tr}\left(RJ_2R'J_3\right)\right)+u_2\operatorname{tr}\left(RJ_1R'J_3\right).
\end{aligned}$$

Remark 2.9. Note that $\dot{V}_1\left(\bar{X},R\right)=\sigma\bar{x}_1\left(1-r_3u_3\right)$ and $\dot{V}_2\left(\bar{X},R\right)=u_1\left(\sigma r_3\bar{x}_3-\operatorname{tr}\left(RJ_2R'J_3\right)\right)+u_2\operatorname{tr}\left(RJ_1R'J_3\right)$. Later it will be important that both quantities will be negative.

Let ε be a (small) positive real number. Set $\xi_1=\sigma r_3\bar{x}_3-\operatorname{tr}\left(RJ_2R'J_3\right)$ and $\xi_2=\operatorname{tr}\left(RJ_1R'J_3\right)$. In order to ensure the non positiveness of $\dot{V}$, we choose smooth feedback controls $u_1=k_1(\bar{X},R)$, $u_2=k_2(\bar{X},R)$ and $u_3=k_3(\bar{X},R)$ such that:

$$\operatorname{sign}k_1(\bar{X},R)=-\operatorname{sign}\xi_1 \tag{2.9}$$

$$\operatorname{sign}k_2(\bar{X},R)=-\operatorname{sign}\xi_2 \tag{2.10}$$

$$k_3(\bar{X},R)=\begin{cases}u_{3\max} & \text{if } \bar{x}_1\geqslant 0\\ -u_{3\max}\leqslant k_3(\bar{X},R)<u_{3\max} & \text{if } \bar{x}_1<0.\end{cases} \tag{2.11}$$

The feedback controls k_1, k_2 may be rewritten as

$$\begin{aligned}
k_1(\bar{X},R)&=-\xi_1\varphi_1\left(\xi_1\right),\\
k_2(\bar{X},R)&=-\xi_2\varphi_2\left(\xi_2\right),
\end{aligned}$$

where φ_1, φ_2 are smooth strictly positive functions and ξ_1 (resp. ξ_2) is zero if and only if $u_1=0$ (resp. $u_2=0$). Hence,

$$\dot{V}\big(\bar{X},R\big)=\sigma\bar{x}_1\left(1-r_3u_3\right)-\xi_1^2\varphi_1\left(\xi_1\right)-\xi_2^2\varphi_2\left(\xi_1\right).$$

According to LaSalle's principle and since V is a proper function, we infer that all the trajectories of system (2.7) with feedback controls k_1, k_2 and k_3 converge to the largest invariant set I contained in the set

$$\begin{aligned}
E&=\{(\bar{X},R)\mid\dot{V}(\bar{X},R)=0\}\\
&=\{(\bar{X},R)\mid k_1(\bar{X},R)=k_2(\bar{X},R)=0,\ k_3(\bar{X},R)=u_{3\max}\}\\
&=\{(\bar{X},R)\mid \xi_1=\xi_2=0,\ k_3(\bar{X},R)=u_{3\max}\}.
\end{aligned} \tag{2.12}$$

Obviously we have $\bar{\mathcal{C}}\subset I$. Let us prove the converse. Due to conditions (2.12) on the feedback controls, it is easy to see that any solution to (2.5) starting from $(\bar{X},R)\in E$ and remaining in E satisfies

$$\begin{cases} \dot{\bar{x}}_1 = u_{3\max}\bar{x}_2 \\ \dot{\bar{x}}_2 = -u_{3\max}\bar{x}_1 \\ \dot{\bar{x}}_3 = 0 \\ \dot{R} = R u_{3\max} J_3. \end{cases} \tag{2.13}$$

Consequently, any solution to (2.13) starting from I with $\bar{x}_1 > 0$ escapes from E in finite time. Thus, any point in I satisfies $\bar{x}_1 = 0$ and consequently, due to the first equation of system (2.13), $\bar{x}_2 = 0$. Moreover, according to the third equation of system (2.13) $\bar{x}_3$ is constant in E. This constant is easily seen to be equal to zero. Indeed, let $t \mapsto \left(\bar{X}(t), R(t)\right)$ be a smooth curve in I defined on an interval around zero. Since the functions ξ_1 and ξ_2 are identically zero in E, we have

$$\sigma r_3 \bar{x}_3 = \operatorname{tr}\left(RJ_2R'J_3\right), \tag{2.14}$$

$$0 = \operatorname{tr}\left(RJ_1R'J_3\right). \tag{2.15}$$

Since $\bar{x}_3$ is constant, differentiation of equation (2.15) (divided by $u_{3\max}$) w.r.t. t yields

$$\begin{aligned} 0 &= \frac{1}{u_{3\max}}\frac{d}{dt}\operatorname{tr}\left(RJ_1R'J_3\right) \\ &= \frac{1}{u_{3\max}}\operatorname{tr}\left(R[u_{3\max}J_3, J_1]R'J_3\right) \\ &= \operatorname{tr}\left(RJ_2R'J_3\right), \end{aligned} \tag{2.16}$$

which, according to (2.14), implies that $\bar{x}_3 = 0$. Summing up, we get $I \subset \{\bar{x}_1 = \bar{x}_2 = \bar{x}_3 = 0\}$. It remains to show that R has the required form. Equations 2.15 and 2.16 may be rewritten as $\operatorname{tr}(J_1(R'J_3R)) = 0$ and $\operatorname{tr}(J_2(R'J_3R)) = 0$ showing that $R'J_3R$ is orthogonal to J_1 and J_2 for the Hilbert-Schmidt scalar product. It follows that $R'J_3R = \lambda J_3$ with $\lambda \in \mathbb{R}$. Moreover, conjugations by elements of $SO(3)$ being isometries for the Hilbert-Schmidt scalar product, we must have $|\lambda| = 1$. Consequently, $R'J_3R = \pm J_3$. The case $\lambda = +1$ exactly means that the rotation matrix R commutes with J_3. According to Lemma 2.10 below, R is of the form $e^{J_3\theta}$. The case $\lambda = -1$ is similar and we conclude that R is of the form $\Delta_1 e^{J_3\theta}$.

It is clear that all trajectories $\mathcal{T}(t)$ starting in the plane $\{\bar{x}_3 = 0\}$ with $R(0) = \Delta_1 e^{J_3\theta}$ tend to $\bar{\mathcal{C}}_2$ since $u_1 \equiv 0$, $u_2 \equiv 0$.

Let us show that any other trajectory tends to $\bar{\mathcal{C}}_1$. Due to Remark 2.9, $\dot{V}_2 \leqslant 0$ then V_2 cannot increase along any trajectory. But V_2 reaches its maximum exactly on the union of the trajectories of type $\mathcal{T}(t)$.

Lemma 2.10. *The stabilizer $\mathcal{S}_3$ of J_3 under conjugation by $SO(3)$ is the subgroup of rotations R_3 about the x_3-axis,* $R_3 = \left(\begin{smallmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{smallmatrix}\right)$.

Proof. With obvious block notations, write $J_3 = \left(\begin{smallmatrix} J & 0 \\ 0 & 0 \end{smallmatrix}\right)$ and $H = \left(\begin{smallmatrix} A & B \\ C & d \end{smallmatrix}\right)$. Then, $[J_3, H] = \left(\begin{smallmatrix} [J,A] & JB \\ -CJ & 0 \end{smallmatrix}\right)$. For $H \in \mathcal{S}_3$, $H'J_3H = J_3$, which is equivalent to $[J_3, H] = 0$. It implies that $B = 0$, $C = 0$ and $[J, A] = 0$. Since $H \in SO(3)$, we first conclude that $d = \pm 1$ and $A \in O(2)$.

The stabilizer of J under conjugation by $O(2)$ is $SO(2)$. Then, $A \in SO(2)$, $\det(A) = 1$, and $d = 1$. Therefore, H is of the form R_3. ∎

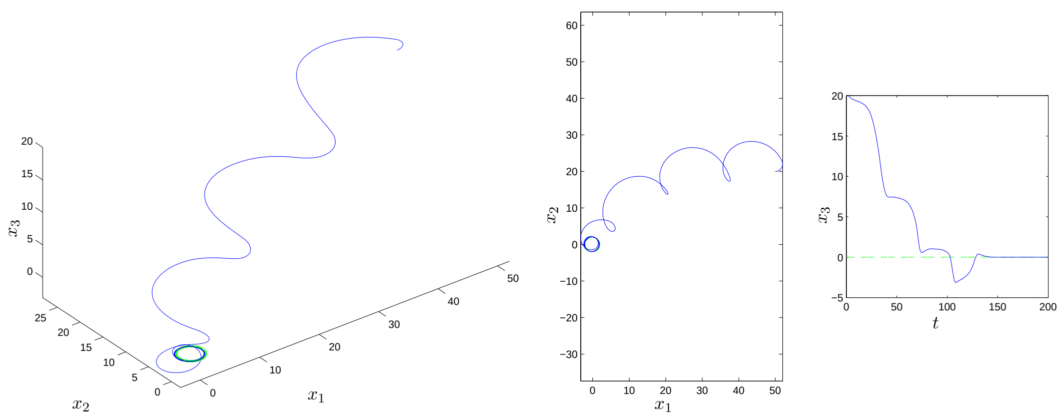
2.2.3 Numerical results

The theory exposed in the preceding subsections was tested by means of simulated data. During these simulations, we numerically integrate the closed-loop system resulting from applying the feedback control $u = (u_1, u_2, u_3) = (k_1(X, R), k_2(X, R), k_3(X, R))$ of Theorem 2.6 to system (2.5) during 200 units of time.

The feedback control k_3 is built as its analogue k in the 2D-case. Concerning the feedback controls k_1 and k_2 we chose smooth functions varying slowly between the minimum and the maximum control constraints.

Figures 2 and 3 display trajectories obtained for the same UAV which starts at the point $(50, 20, 20)$ with a direction of 0 radian for the pitch, roll and yaw angles. We consider the following curvature constraints, $u_{1\,\max} = u_{2\,\max} = -u_{1\,\min} = -u_{2\,\min} = 0.1$, $u_{3\,\max} = 0.5$, $u_{3\,\min} = 0.1$.

The difference between Figure 2 and Figure 3 is due to the value of the parameter ε in the control function defined in equation (2.11). In the first case we set $\varepsilon = 3$ and we use the value $\varepsilon = 1$ in the second simulation. In each case we set $\sigma = 1$ in (2.8).



(a) (b)

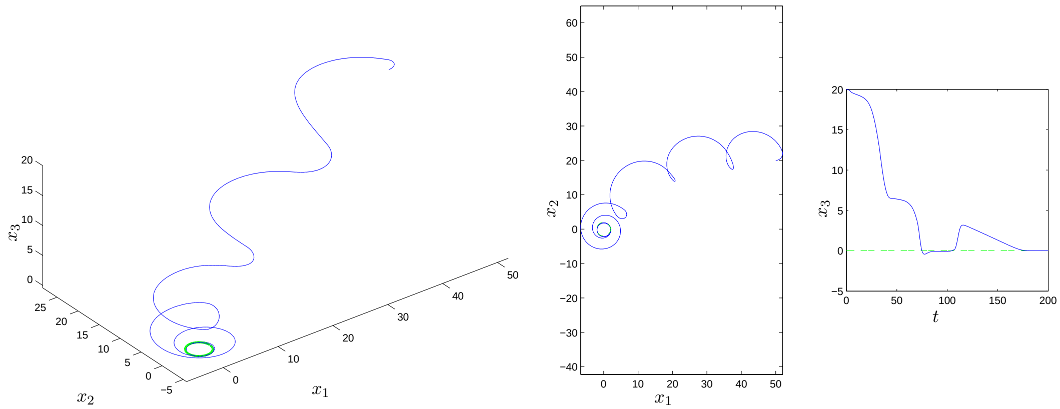
Figure 2: Illustration of Lyapunov-LaSalle-based 3D-planification results with starting point $(x_{10}, x_{20}, x_{30}) = (50, 20, 20)$, control constraints $u_{1\,\max} = u_{2\,\max} = 0.1$, $u_{3\,\max} = 0.5$, $a_3 = 0.1$, $\sigma = 1$ and $\varepsilon = 3$. (a) The $(x_1(\cdot), x_2(\cdot), x_3(\cdot))$ trajectory, (b) projection in the horizontal plane (left) and evolution of x_3 w.r.t. time (right).

3 The time-optimal stabilizing synthesis

In this section we study the minimum time problem for system 2.1. For simplicity of exposition we assume that $[-u_{\max}, u_{\max}] = [-1, 1]$ (cf. Remark 2.1). More precisely, we consider the following problem that we shall denote by **(P)**.

(P) For every $(x_0, y_0, \theta_0) \in \mathbb{R}^2 \times S^1$ find the pair trajectory-control joining (x_0, y_0, θ_0) to $\mathcal{C}$, which is time-optimal for the control system

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{\theta} = u, \quad u \in [-1, 1]. \end{cases} \tag{3.1}$$



(a)

(b) Projection in the horizontal plan and evolution of x_3 over the time.

Figure 3: Illustration of Lyapunov-LaSalle-based 3D-planification results with $\sigma = 1$ and $\varepsilon = 1$.

To solve Problem **(P)** it is again convenient to work with a reduced system in dimension two. Indeed, in dimension two, a complete theory for finding the time-optimal synthesis exists and will be recalled in the next two sections. In this section we make the following transformation (in $O(2) \setminus SO(2)$):

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

This transformation decouples the variable θ and the final manifold C projects to the point $(\tilde{x}, \tilde{y}) = (0, 1)$. We thus consider the following problem that we shall denote by **(P′)**.

(P′) For every $(\tilde{x}_0, \tilde{y}_0) \in \mathbb{R}^2$ find the pair trajectory-control joining $(\tilde{x}_0, \tilde{y}_0)$ to $q_0 = (0, 1)$, which is time-optimal for the control system

$$\begin{cases} \dot{\tilde{x}} = -u\tilde{y} + 1 \\ \dot{\tilde{y}} = u\tilde{x} \end{cases}, \quad u \in [-1, 1].$$

The collection of all solutions to Problem **(P′)** for every $(\tilde{x}_0, \tilde{y}_0)$ is called the *time-optimal stabilizing synthesis* ([37]).

First, it is convenient to change the sign of the dynamics and to consider the equivalent problem (denoted by **(Q)**) of finding the *time-optimal synthesis* issued from q_0:

(Q) For every $(\tilde{x}_f, \tilde{y}_f) \in \mathbb{R}^2$ find the pair trajectory-control joining $q_0 = (0, 1)$ to $(\tilde{x}_f, \tilde{y}_f)$, which is time-optimal for the control system

$$\begin{cases} \dot{\tilde{x}} = u\tilde{y} - 1 \\ \dot{\tilde{y}} = -u\tilde{x} \end{cases}, \quad u \in [-1, 1]. \tag{3.2}$$

Once that Problem **(Q)** is solved, then the time-optimal stabilizing synthesis is obtained simply by inverting the arrows of the trajectories of the time-optimal synthesis.

3.1 Basic definitions and Pontryagin Maximum Principle

Let M be a n-dimensional connected smooth manifold and let F and G be smooth vector fields on M. Consider the following general control-affine time-optimal problem that we shall denote by **(R)**.

(R) For every q_0 and q_f in M find the pair trajectory-control joining q_0 to q_f, which is time-optimal for the control system

$$\dot{q} = F(q) + uG(q), \quad q \in M, \quad u \in [-1, 1]. \tag{3.3}$$

Definition 3.1 (admissible control/trajectory)**.** An *admissible control* $u(\cdot)$ for the system (3.3) is a measurable function $u(\cdot) : [a, b] \to [-1, 1]$. An *admissible trajectory* is a Lipschitz function $q(\cdot) : [a, b] \to M$ satisfying $\dot{q}(t) = F(q(t)) + u(t)G(q(t))$ a.e. for some admissible control $u(\cdot)$.

In the following we assume that the control system is *complete* i.e., for every measurable control function $u(\cdot) : [a, b] \to [-1, 1]$ and every initial state q_0, there exists a trajectory $q(\cdot)$ corresponding to $u(\cdot)$, which is defined on the whole interval $[a, b]$ and satisfies $q(a) = q_0$.

Thanks to the compactness of the set of controls, the convexity of the set of velocities and the completeness of the control system, Filippov's theorem (see for instance [1]) gives,

Proposition 3.2. *For any pair of points in M, there exists a time-optimal trajectory joining them.*

The main tool to compute time-optimal trajectories is the Pontryagin's Maximum Principle (PMP for short). We refer to [1] for a general version of PMP and also [12] for a version of PMP on two-dimensional manifolds for single input control affine systems. The PMP is a first order necessary condition for optimality.

The following theorem is a version of PMP for control systems of the form (3.3) that we state in our own context only.

Theorem 3.1 (PMP for problem **(R)**)**.** *Consider the control system (3.3). Define for every $(q, p, u) \in T^*M \times [-1, 1]$ the function*

$$H(q, p, u) = \langle p, F(q)\rangle + u \langle p, G(q)\rangle . \tag{3.4}$$

If the pair $(q(\cdot), u(\cdot)) : [0, T] \to M \times [-1, 1]$ is time-optimal then there exists a never-vanishing Lipschitz covector $p(\cdot) : t \in [0, T] \mapsto p(t) \in T^*_{q(t)}M$ and a constant $\lambda \leqslant 0$ such that for a.e. $t \in [0, T]$:

i. $\dot{q}(t) = \dfrac{\partial H}{\partial p}(q(t), p(t), u(t))$,

ii. $\dot{p}(t) = -\dfrac{\partial H}{\partial q}(q(t), p(t), u(t))$,

iii. $H(q(t), p(t), u(t)) = \max\limits_{u\in[-1,1]} H(q, p, u)$,

iv. $H(q(t), p(t), u(t)) + \lambda = 0$.

Remark 3.3. A trajectory $q(\cdot)$ (resp. a pair $(q(\cdot), p(\cdot))$) satisfying the conditions given by the PMP is said to be an *extremal* (resp. an *extremal pair*). An extremal corresponding to $\lambda = 0$ is said to be an *abnormal extremal*, otherwise we call it a *normal extremal*.

3.2 An overview on optimal synthesis on 2D-manifolds

In this section we briefly recall the theory of time-optimal syntheses on 2D-manifolds for systems of the form (3.3), developed by Sussmann, Bressan, Piccoli and Boscain in [11, 13, 36, 46] and rewritten in [12].

This section is written to be as much self-consistent as possible. In the remainder of the section, we assume M to be of dimension two.

3.2.1 Basic definitions

For every coordinate chart on the manifold M it is possible to introduce the following three functions:

$$\Delta_A(q) = \det(F(q), G(q)), \tag{3.5}$$

$$\Delta_B(q) = \det(G(q), [F, G](q)), \tag{3.6}$$

and on $M \setminus \Delta_A^{-1}(0)$

$$f_S(q) = -\frac{\Delta_B(q)}{\Delta_A(q)}. \tag{3.7}$$

The sets $\Delta_A^{-1}(0)$, $\Delta_B^{-1}(0)$ of zeros of Δ_A, Δ_B are respectively the set of points where F and G are parallel, and the set of points where G is parallel to $[F, G]$. These loci are fundamental in the construction of the optimal synthesis. In fact, following [12], assuming that they are smooth embedded one dimensional submanifolds of M we have the following:

- on each connected component of $M \setminus (\Delta_A^{-1}(0) \bigcup \Delta_B^{-1}(0))$, every extremal trajectory is bang-bang with at most one switching. Moreover, for every switching of the extremal trajectory the value of the control switches from -1 to $+1$ if $f_S > 0$ and from $+1$ to -1 if $f_S < 0$;
- the support of singular trajectories (i.e., trajectories along which the switching function vanishes identically, see Definition 3.5 below) is always contained in the set $\Delta_B^{-1}(0)$.

Then, the synthesis is built recursively w.r.t. the number of switchings of extremal trajectories, canceling at each step the non optimal trajectories (see [12, Chapter 1]).

In the optimal syntheses some special curves appears, namely:

- Switching curves, i.e., curves made of points where the control switches from $+1$ to -1 or vice versa.
- Singular curves, i.e., curves along which the corresponding switching function (see Definition 3.5 below) is identically zero. For these curves the corresponding control can take values in the interior of $[0, 1]$.
- Cut loci, namely curves made of cut points. Given an admissible curve γ of the control system, defined on $[0, T]$, we say that $t_{\text{cut}} \in (0, T)$ (resp. $\gamma(t_{\text{cut}})$) is a cut time (resp. point) if $\gamma|_{[0,t_{\text{cut}}]}$ is time-optimal and $\gamma|_{[0,t_{\text{cut}}+\varepsilon]}$ is not for every sufficiently small ε.

Remark 3.4. Notice that, although the functions Δ_A and Δ_B depend on the coordinate chart, the sets $\Delta_A^{-1}(0)$, $\Delta_B^{-1}(0)$ and the function f_S do not, i.e., they are intrinsic objects related with the control system (3.3).

We are now interested in determining the extremals. A key role is played by the following:

Definition 3.5 (switching function)**.** Let $(q(\cdot), p(\cdot))$ be an extremal pair. The corresponding switching function is defined as $\phi(t) = \langle p(t), G(q(t))\rangle$.

Notice that $\phi(\cdot)$ is continuously differentiable since $\dot{\phi}(t) = \langle p(t), [F, G](q(t))\rangle$ which is continuous.

Definition 3.6 (bang, singular)**.** Let $q(\cdot)$ be an extremal trajectory defined on the time interval $[a, b]$, and let $u(\cdot) : [a, b] \to [-1, 1]$ be the corresponding control. We say that $u(\cdot)$ is a *bang* control if $u(t) = +1$ a.e. in $[a, b]$ or $u(t) = -1$ a.e. in $[a, b]$. We say that $u(\cdot)$ is *singular* if the corresponding switching function $\phi(t) = 0$ in $[a, b]$. A finite concatenation of bang controls is called a *bang-bang* control. A *switching time* of $u(\cdot)$ is a time $\tau \in [a, b]$ such that, for every $\varepsilon > 0$, $u(\cdot)$ is not bang or singular on $(\tau-\varepsilon, \tau+\varepsilon)\bigcap[a, b]$. An extremal trajectory of the control system (3.3) is called a bang extremal, singular extremal, bang-bang extremal respectively, if it corresponds to a bang control, singular control, bang-bang control respectively. If τ is a switching time, the corresponding point $q(\tau)$ on the trajectory $q(\cdot)$ is called a *switching point.*

The switching function is important because it determines where the controls may switch. In fact, using the PMP, one easily gets:

Proposition 3.7 ([12, Lemma 5 page 40])**.** *A necessary condition for a time τ to be a switching time is that $\phi(\tau) = 0$. Therefore, on any interval where ϕ has no zeros (respectively finitely many zeros), the corresponding control is bang (respectively bang-bang). In particular, $\phi > 0$ (resp. $\phi < 0$) on $[a, b]$ implies $u = 1$ (resp. $u = -1$) a.e. on $[a, b]$. On the other hand, if ϕ has a zero at τ and $\dot{\phi}(\tau)$ is different from zero, then τ is an isolated switching.*

3.2.2 More on singular extremals and predicting switchings for 2D-systems

In this section we compute the control corresponding to singular extremals and we would like to predict which kind of switchings can happen, using properties of the vector fields F and G. The following lemmas illustrate the role of the set $\Delta_B^{-1}(0)$ in relation with singular extremals. Proofs can be found in [11, 12, 36].

We denote by $\mathrm{Supp}\,(q(\cdot))$ the support of the trajectory $q(\cdot)$.

Lemma 3.8 ([12, Lemma 10 page 46])**.** *Let $q(\cdot)$ be an extremal trajectory which is singular on a time interval $[a, b]$ included in the domain of $q(\cdot)$. Then, $q(\cdot)|_{[a,b]}$ corresponds to the so-called singular control $\varphi(q(t))$, where*

$$\varphi(q) = -\frac{d_q\Delta_B\,(F(q))}{d_q\Delta_B\,(G(q))}, \tag{3.8}$$

with Δ_B defined in equation (3.6) and $d_q\Delta_B$ being the differential map of Δ_B at point q. Moreover, on $\mathrm{Supp}\,(q(\cdot))$, $\varphi(q)$ is always well-defined and its absolute value is smaller or equal to one. Finally $\mathrm{Supp}\,(q(\cdot)|_{[a,b]}) \subset \Delta_B^{-1}(0)$.

The following lemma describes what happens when Δ_A and Δ_B are different from zero.

Lemma 3.9 ([12, Theorem 11 page 44])**.** *Let $\Omega \subset M$ be an open connected set such that $\Omega\bigcap(\Delta_A^{-1}(0)\bigcup\Delta_B^{-1}(0)) = \emptyset$. Then all connected components of $\mathrm{Supp}\,(q(\cdot))\bigcap\Omega$, where $q(\cdot)$ is an extremal trajectory of (3.3), are bang-bang with at most one switching. Moreover, if $f_S|_\Omega > 0$, then $q(\cdot)|_\Omega$ is associated to a constant control equal to $+1$ or -1 or has a switching from -1 to $+1$. If $f_S|_\Omega < 0$, then $q(\cdot)|_\Omega$ is associated to a constant control equal to $+1$ or -1 or has a switching from $+1$ to -1.*

Remark 3.10. For the problem **(R)**, under generic conditions on the vector fields F and G, one can make the complete classification of synthesis singularities, stable synthesis, singularities of the minimum time wave fronts. We refer to [12] for the general theory. For some extensions to higher dimension, see [2, 41].

3.3 Construction of the time-optimal stabilizing synthesis for the reduced system

Let us apply the theory of the previous section to problem **(Q)**.

3.3.1 Preliminary observations

Define

$$F = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, \quad G = \begin{pmatrix} \tilde{y} \\ -\tilde{x} \end{pmatrix}, \quad Y = F + G = \begin{pmatrix} -1 + \tilde{y} \\ -\tilde{x} \end{pmatrix}, \quad X = F - G = \begin{pmatrix} -1 - \tilde{y} \\ \tilde{x} \end{pmatrix}.$$

The integral curves of the vector fields Y and X are shown below (Fig. 4).

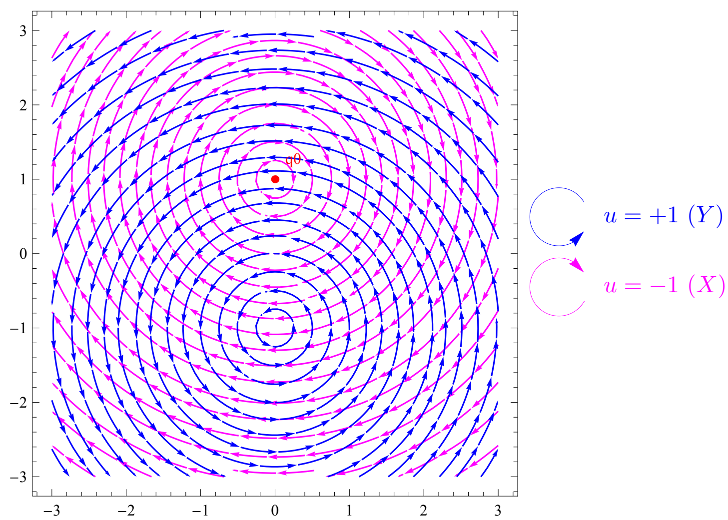


Figure 4: Integral curves of X and Y

We compute the quantities,

$$\begin{aligned}
\Delta_A(\tilde{x}, \tilde{y}) &= \det(F(\tilde{x}, \tilde{y}), G(\tilde{x}, \tilde{y})) = \det \begin{pmatrix} -1 & \tilde{y} \\ 0 & -\tilde{x} \end{pmatrix} = \tilde{x}, \\
\Delta_B(\tilde{x}, \tilde{y}) &= \det(G(\tilde{x}, \tilde{y}), [F, G](\tilde{x}, \tilde{y})) = \det \begin{pmatrix} \tilde{y} & 0 \\ -\tilde{x} & 1 \end{pmatrix} = \tilde{y}, \\
f_S(\tilde{x}, \tilde{y}) &= -\frac{\Delta_B(\tilde{x}, \tilde{y})}{\Delta_A(\tilde{x}, \tilde{y})} = -\frac{\tilde{y}}{\tilde{x}}.
\end{aligned}$$

Using Lemmas 3.8 and 3.9 it follows that:

- on the set $\{\tilde{x} > 0\} \bigcap \{\tilde{y} > 0\}$ only the switching from $+1$ to -1 is allowed. And cyclically on the other quadrants.

- If there are singular trajectories then they should lie on the set $\{\tilde{y} = 0\}$. Notice that a trajectory whose support is $\{\tilde{y} = 0\}$ [2] is admissible (it corresponds to the zero control). See Fig. 5.

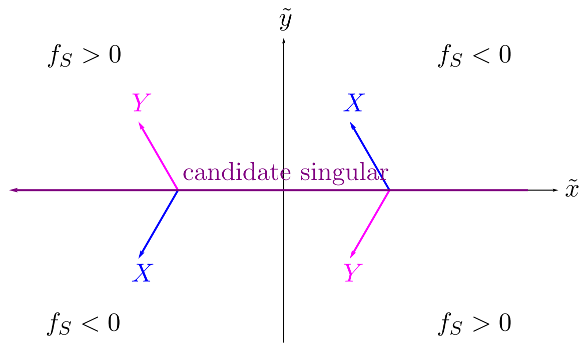


Figure 5: Candidate singular

We construct the time-optimal synthesis in the following way:

- STEP 0. We consider the trajectory γ^- starting from q_0 with control -1. All extremal trajectories should start in this way since in a small neighborhood V of q_0 we have that $f_S < 0$ and hence neither singular trajectories nor trajectories having chattering, i.e. for instance an infinite number of switchings in finite time, could be extremal in V. Moreover, a trajectory cannot start from q_0 with control $+1$ since the vector field $Y(q_0) = 0$.

- STEP 1. We compute the last time at which the trajectory γ^- is extremal (or has lost optimality because it self-intersects) and we study which kind of extremal trajectories can bifurcate from γ^-.

- STEP 2. For each bang or singular trajectory bifurcating from γ^-, we study the last time at which it is extremal. If there are intersections among trajectories, we cancel those parts that are not optimal (among trajectories already computed up to this step).

- STEP 3. For each trajectory γ defined on $[0, \tau_\gamma]$ computed at the previous step do the following. If at τ this trajectory loses optimality (among trajectories of the previous step), do nothing otherwise prolong γ with the next bang or singular trajectory up to the last time at which it is extremal. If there are intersections among trajectories, cancel those parts that are not optimal (among trajectories already computed up to this step). Notice that in general there could be several prolongations of γ that are extremals (due to the presence of singular trajectories, see [12, §2.6.4 page 62]). Hence, in general, new extremal trajectories are generated at this step.

Then, the synthesis is built recursively, repeating STEP 3 up to when no new trajectories are generated. Notice that a trajectory generated at STEP n has n concatenations of bang or singular trajectories. In our case, only one application of STEP 3 is necessary.

[2] According to [12], the set $\{\tilde{y} = 0\}$ is a turnpike (i.e., a trajectory that is locally time-optimal)

3.3.2 Adjoint equation

To compute the last time at which bang and singular trajectories are extremal, we need the adjoint equation. From equation (3.4) we have that $H(\tilde{x}, \tilde{y}, \xi, \zeta, u) = -\xi + u(\xi\tilde{y} - \zeta\tilde{x})$ (here $q = (\tilde{x}, \tilde{y})$ and $p = (\xi, \zeta)$). Hence,

$$\begin{cases} \dot{\xi} = u(t)\zeta \\ \dot{\zeta} = -u(t)\xi. \end{cases} \tag{3.9}$$

3.3.3 STEP 1: Analysis along the curve starting from q_0 and corresponding to control -1

The trajectory γ^- starting from q_0 and corresponding to control -1 has coordinates:

$$\begin{aligned} \tilde{x}^-(t) &= -2\sin(t) \\ \tilde{y}^-(t) &= 2\cos(t) - 1 \end{aligned}$$

This curve intersects the $\tilde{x}$-axis at time $t_1 = \pi/3$. The solution of the adjoint equation with $u = -1$ and the normalization $|p(0)| = 1$ is,

$$\begin{aligned} \xi(t, \alpha) &= \cos(t + \alpha) \\ \zeta(t, \alpha) &= \sin(t + \alpha), \end{aligned}$$

where α is defined by $\cos(\alpha) = \xi(0)$, $\sin(\alpha) = \zeta(0)$. Notice that the condition $H + \lambda = -\xi + u(\xi\tilde{y} - \zeta\tilde{x}) + \lambda = 0$ with $\lambda \leqslant 0$, written at the initial point implies that,

$$-\cos(\alpha) + u\cos(\alpha) = \cos(\alpha)(u - 1) \geqslant 0.$$

Since we start with control -1 we have that $\alpha \in [\pi/2, 3/2\pi]$.

Then, the switching function is given by

$$\phi_\alpha(t) = \langle p(t), G(\gamma^-(t)) \rangle = 2\cos(\alpha) - \cos(t + \alpha).$$

By studying this function it follows that:

- for $\alpha = \pi/2$, ϕ_α starts with value zero and then takes positive values: it cannot correspond to a trajectory starting with control -1.

- For $\alpha \in (\pi/2, 2\pi/3)$, ϕ_α starts with negative values, and has its first zero (of order one) (close to the origin for values of α close to $\pi/2$). For every $\tau \in (0, \pi/3)$ there exists a unique $\alpha \in (\pi/2, \pi/3)$ for which the switching function has a zero at τ.

- For $\alpha = 2\pi/3$, ϕ_α starts with negative values and has a double zero at $t = \pi/3$ (the point q_S at which the trajectory meets the turnpike). Notice that there exists an extremal trajectory entering the turnpike, since there is a trajectory switching there (see the singularity (Y, S) in [12, page 62]).

- For $\alpha \in (2\pi/3, 4\pi/3)$, ϕ_α is always negative. This means that the trajectory with control -1 is always extremal (a priori, γ^- can make many turns !).

- For $\alpha = 4\pi/3$, ϕ_α starts with negative values and has a double zero at $t = 5\pi/3$ (again at a point $\bar{q}_S$ where the trajectory meets the turnpike). Again notice that there exists an extremal trajectory entering the turnpike, since there is a trajectory switching there.

- For $\alpha \in (4\pi/3, 3\pi/2)$, ϕ_α starts with negative values, and then has its first zero (of order one) (close to $t = 5\pi/3$ for small values of α close $4\pi/3$). For every $\tau \in (\pi, 5\pi/3)$ there exists a unique $\alpha \in (4\pi/3, 3\pi/2)$ for which the switching function has a zero at τ.
- For $\alpha = 3\pi/2$, ϕ_α starts with negative values, and has its first zero (of order one) at $t = \pi$.

This study being done, we get the expression of the first switching time τ_1 solving the equation $\phi_\alpha(\tau_1) = 0$ (and taking into account that $\tau_1 > 0$). It yields

$$\tau_1(\alpha) = -\alpha + \arccos(2\cos\alpha), \quad \alpha \in \left(\frac{\pi}{2}, \frac{2\pi}{3}\right] \bigcup \left[\frac{4\pi}{3}, \frac{3\pi}{2}\right]. \tag{3.10}$$

Notice that γ^- is extremal for every $t > 0$. However, γ^- cannot be optimal after $t = 2\pi$ since it is periodic.

The initial conditions of the covector are shown below (Fig. 6).

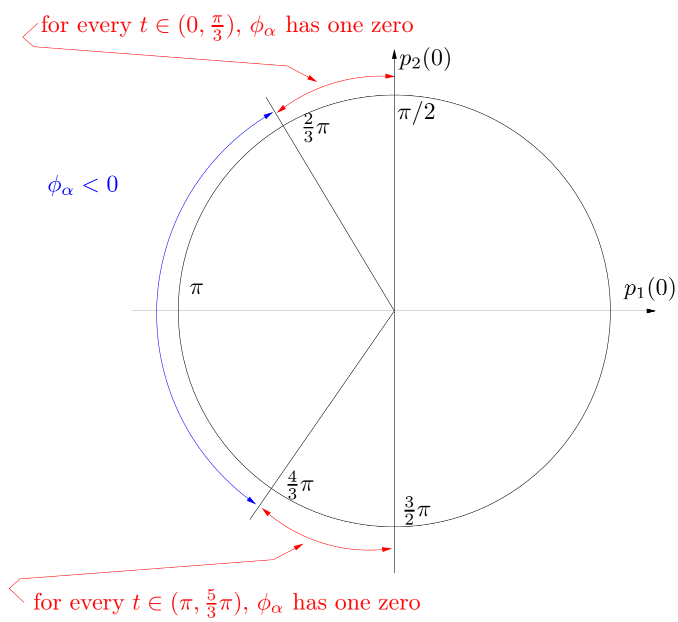


Figure 6: Initial conditions of covector

3.3.4 STEP 2

At this step we study separately the following four families of trajectories bifurcating from γ^-.

- Family 1: trajectories bifurcating from $\gamma^-[0, \pi/3)$.
- Family 2: the singular trajectory bifurcating from $q_S = \gamma^-(\pi/3) = (-\sqrt{3}, 0)$.
- Family 3: trajectories bifurcating from $\gamma^-[\pi, 5/3\pi]$
- Family 4: the singular trajectory bifurcating from $\bar{q}_S = \gamma^-(5\pi/3) = (\sqrt{3}, 0)$

The extremal trajectories bifurcating from γ^- are drawn in the following picture (Fig. 7).

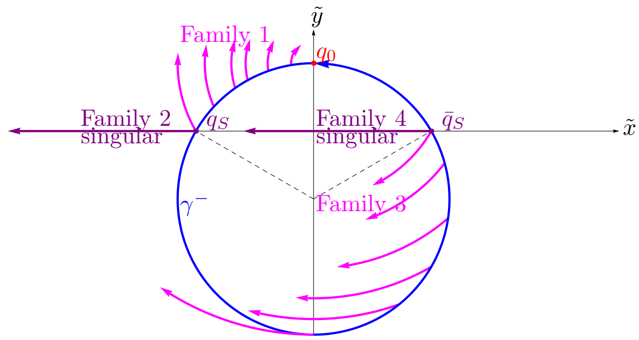


Figure 7: Extremal trajectories bifurcating from γ^-

Family 1. Consider a curve corresponding to control $+1$ starting from the point $\gamma^-(\tau_1)$, with $\tau_1 = \tau_1(\alpha)$, $\alpha \in [\pi/2, 2\pi/3]$ (which corresponds to $\tau_1 \in [0, \pi/3]$). Its coordinates are:

$$\begin{aligned} x(t,\tau_1) &= 2\sin(t-\tau_1) - 2\sin(t) \\ y(t,\tau_1) &= 1 - 2\cos(t) + 2\cos(t-\tau_1) \end{aligned}$$

The corresponding covector which satisfies the switching condition at the beginning

$$p(0,\tau_1)G(x(0,\tau_1), y(0,\tau_1)) = 0,$$

has coordinates

$$\begin{aligned} \xi(t,\tau_1) &= \cos(t-\tau_1-\alpha) \\ \zeta(t,\tau_1) &= -\sin(t-\tau_1-\alpha), \end{aligned}$$

The switching function is given by (here we have taken account that $\phi_\alpha(\tau_1) = 0$)

$$\tilde{\phi}_\alpha(t+\tau_1) = \cos(t-\tau_1-\alpha) - 2\cos(\alpha).$$

We get the expression of the second switching time τ_2 solving the equation $\tilde{\phi}_\alpha(\tau_2+\tau_1) = 0$ (and taking into account that $\tau_2 > 0$). It yields

$$\tau_2(\alpha) = 2\arccos(2\cos\alpha), \quad \alpha \in \left(\frac{\pi}{2}, \frac{2\pi}{3}\right]. \tag{3.11}$$

The switching curve is shown in the following picture (Fig. 8).

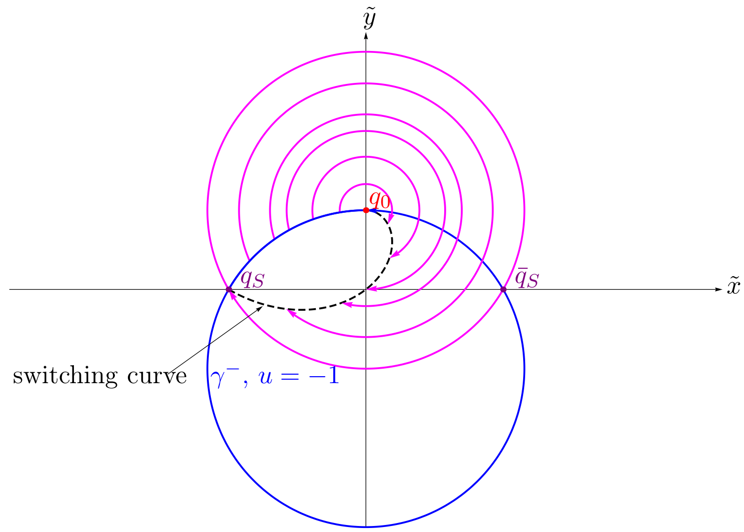


Figure 8: Switching curve family 1

Family 2: the singular trajectory. Since there is a trajectory switching from γ^- in q_S it follows that there exists an extremal trajectory entering the turnpike. At the entrance point of the turnpike the covector satisfies $\langle p_S, G(q_S)\rangle = 0$. Since $G = (y, -x)$ it follows that $\left\langle p_S, (0, \sqrt{3})\right\rangle = 0$. Hence we can assume that

$$p_S = (-1, 0).$$

The minus sign has been chosen in such a way that $\langle p_S, F + uG\rangle \geqslant 0$ (according to Theorem 3.1, $H + \lambda = \langle p_S, F + uG\rangle + \lambda = 0$, with $\lambda \leqslant 0$). On the singular trajectory the control can be only zero. From equation (3.9) it follows that p is constant. Hence the singular trajectory is extremal for all positive times (Fig. 9).

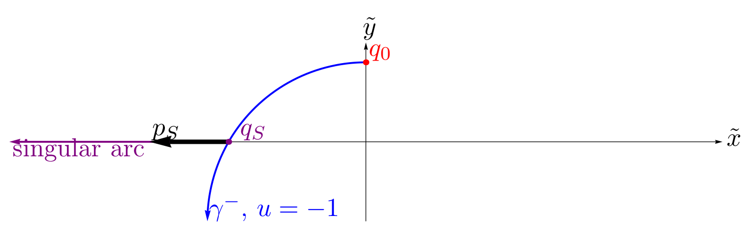


Figure 9: Singular arc

Family 3. This family is treated exactly as Family 1. The only difference is that now $\alpha \in [4\pi/3, 3\pi/2]$ (which corresponds to $\tau_1 \in [\pi, 5\pi/3]$). The switching curve is shown on the following picture (Fig. 10).

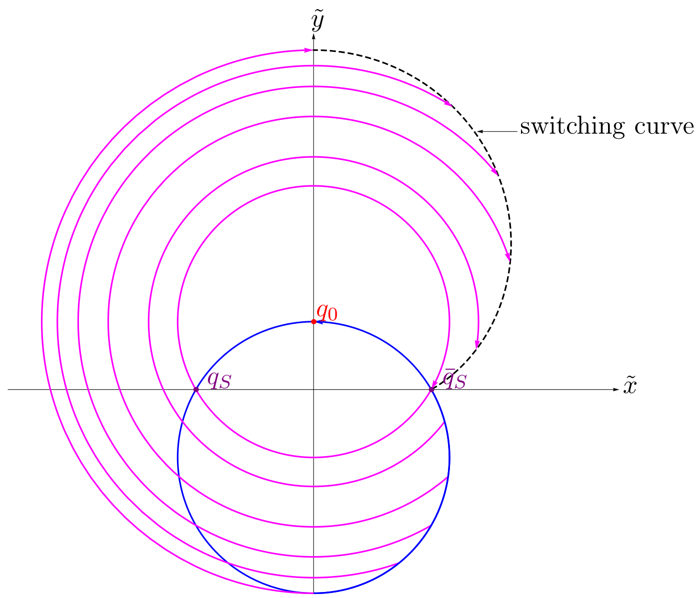


Figure 10: Switching curve family 3

Family 4. This family is treated exactly as Family 2. Since there is a trajectory switching from γ^- at $\bar{q}_S$ it follows that there exists an extremal trajectory entering the turnpike. This trajectory is then extremal for every positive time (Fig. 11).

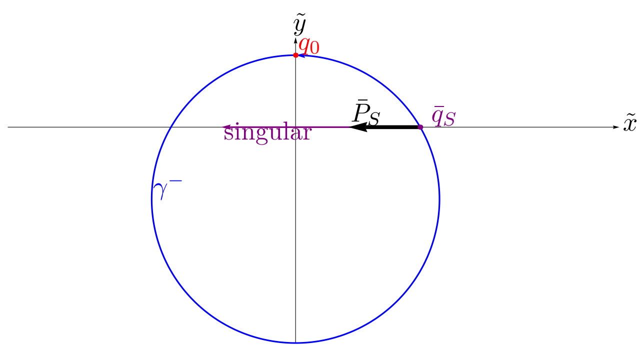


Figure 11

Cutting non optimal trajectories. A direct computation yields the following.

- The trajectory γ^- is not optimal after passing through the point $\bar{q}_S$.
- Let us call γ_{crazy} the trajectory of Family 3 corresponding to $\tau_1 = \pi$ (i.e., $\alpha = 3\pi/2$). This trajectory is not optimal after crossing the singular trajectory of Family 2.
- The trajectories of Family 3 for $\tau_1 \in (\pi, 5\pi/3)$ (i.e., $\alpha \in (4\pi/3, 3\pi/2)$) are not optimal after crossing the γ^- trajectory.
- Let us call γ_{double} the trajectory of Family 3 corresponding to $\tau_1 = 5\pi/3$ (i.e., $\alpha = 4\pi/3$). This trajectory is not optimal after passing through the point q_S. Notice that the support of this trajectory coincides with part of the support of a trajectory of Family 1 (the one corresponding to $\tau_1 = \pi/3$ (i.e., $\alpha = 2\pi/3$).

- The singular trajectory of Family 4 is not optimal.

It follows that only the trajectories shown in the following picture are candidates to be optimal. These are the trajectories generated at STEP 2 (Fig. 12).

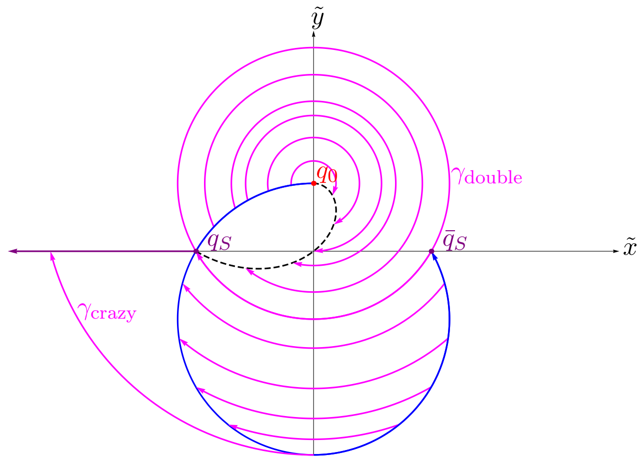


Figure 12: Trajectories generated at STEP 2

3.3.5 STEP 3

The trajectories generated at this step are shown on the following figure where they are divided in four families: family 2.1, 2.2, 2.3 and 2.4 (Fig. 13).

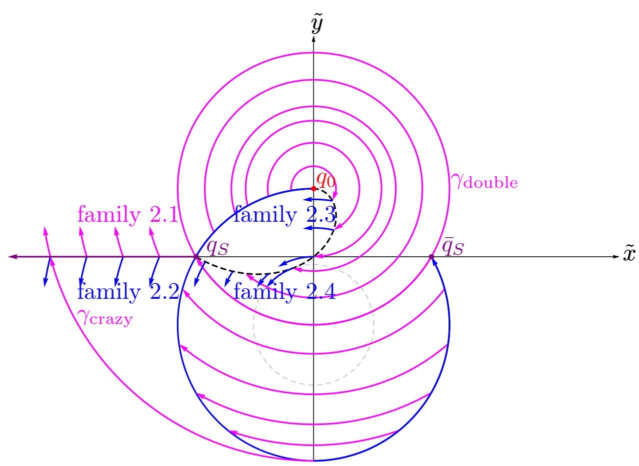


Figure 13: Trajectories generated at STEP 3

The point q_{top} that divides the families 2.3 and 2.4 is the point at which the switching curve generated at the previous step become tangent to an integral curve of the vector field X.

The coordinates of the point q_{top} are $\left(-\sqrt{\frac{59\sqrt{3}}{2}-51}, \frac{1}{2}\left(5\sqrt{3}-9\right)\right)$.

Remark 3.11. Notice the following important fact. The trajectory of Family 1 of STEP 1 corresponding to $\alpha = \arccos\left(-\sqrt{5}/4\sqrt{2}\right)$ (i.e., $\tau_1 = 2\arcsin(1/4)$ and $\tau_2 = 3\pi/2 + \arcsin(1/4)$) has a switching at the origin at time $t = 3\pi/2 + 3\arcsin(1/4)$. However, this trajectory cannot be connected to a singular trajectory. This is due to the fact that the vector field G is zero at this point. Hence, the switching function vanishes but the covector has not necessary the right direction to make the trajectory enter the turnpike extremal. This phenomenon is similar to the one described in [10, Fig. 4.6]. A direct computation shows that the value of the covector of this trajectory at the origin is $\left(-\sqrt{5/8}, \sqrt{3/8}\right)$ while a covector of the form $(-1, 0)$ is necessary to enter the turnpike.

We have the following

- The family 2.4 is clearly not optimal since it is generated by a switching curve that *reflects* the trajectories (see the analysis of $\bar{C}$ curves in [12, §4.1.3]).
- The families 2.1 and 2.2 do not stop to be extremal before meeting the set $\mathcal{K}_1 = \{(x, 0) | x > \sqrt{3}\}$. This set is a cut locus. The curve γ_{crazy} is not optimal. It is quicker to take trajectories from the family 2.2.
- The family 2.3 loses optimality when it meets Family 3 of the previous step. It forms then a cut locus $\mathcal{K}_2$ which is a one-dimensional manifold starting from q_{top} and arriving to a point q_{bottom} on the γ^- trajectory. The coordinates of the point q_{bottom} are $\left(-\sqrt{3}, -2\right)$.

  This cut locus has the following features:

  – at q_{top} it is C^1-connected to the switching curve from which the family 2.3 is generated. This tangency is a consequence of the general theory developed in [12]. It is called a $(C, K)_2$ singularity (see [12, Fig. 2.21]).

  – It is not smooth at one point that we denote by q_{middle}. More precisely, it is not smooth at the intersection with the curve γ_{double} that separates the families 1 and 3 of the previous STEP. The presence of this nonsmoothness point is a consequence of the fact that the minimum-time front is not smooth along the curve γ_{double}. Indeed this trajectory separates two different families.
  The coordinates of the point q_{middle} are $\left(-\frac{\sqrt{15}}{4}, -\frac{3}{4}\right)$.

  – At the point q_{bottom} it is C^1-connected to the curve γ^-. This fact is the consequence of the tangency of Family 3 with γ^- at the point $\gamma^-(\pi)$.

Since all the trajectories generated at this step are either defined on $[0, +\infty)$ or on $[0, T]$ with cut time T, it follows that no new trajectories are generated at subsequent steps. Hence, the time-optimal synthesis is constructed (Fig. 14).

Remark 3.12. Notice that the minimum time function is not continuous on $\gamma^-([0, \pi))$.

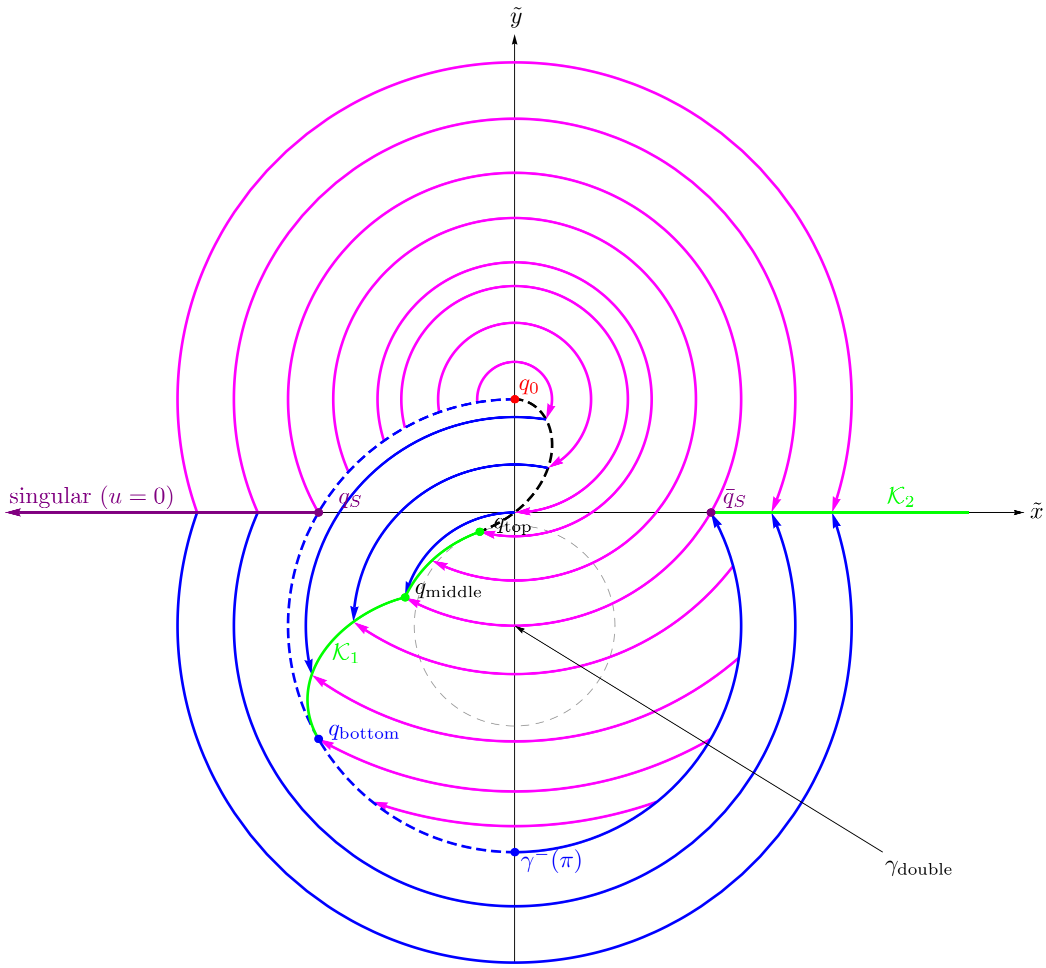


Figure 14: The optimal synthesis. All optimal trajectories start at q_0 with control -1. The dashed black curve is the switching curve, the purple curve is the singular trajectory and the green curves are cut loci. Notice that the minimum-time function is not continuous along the blue dashed curve.

3.4 Correspondance between the solutions to problem (P) and problem (Q)

The solutions to problem **(P)** can be deduced from the solutions to problem **(Q)**. In this section we display pairs of figures showing a solution of problem **(P)** and its corresponding lifted solution to problem **(Q)**. Pictures are shown for each type of solution: bang-bang, bang-bang-bang, and bang-singular-bang trajectories.

Let us make a few remark on the time-optimal synthesis of problem **(P)**.

As the reader can easily check using a similar argument as it was done in [44], the application of PMP to problem **(P)** leads to the following lemma describing some of the feature of the synthesis.

Lemma 3.13. *Optimal paths solving Problem* **(P)** *are such that rectilinear segments and points of inflection belong to a same line passing through the center of the final manifold* $\mathcal{C}$*.*

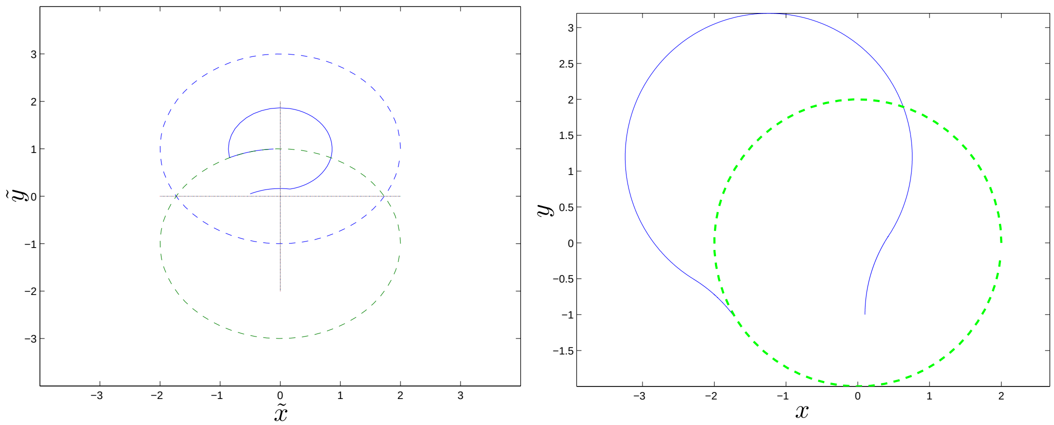


Figure 15: A bang-bang-bang optimal trajectory. A bang-bang-bang solution to problem **(Q)** (left) and the corresponding optimal solution to problem **(P)** (right).

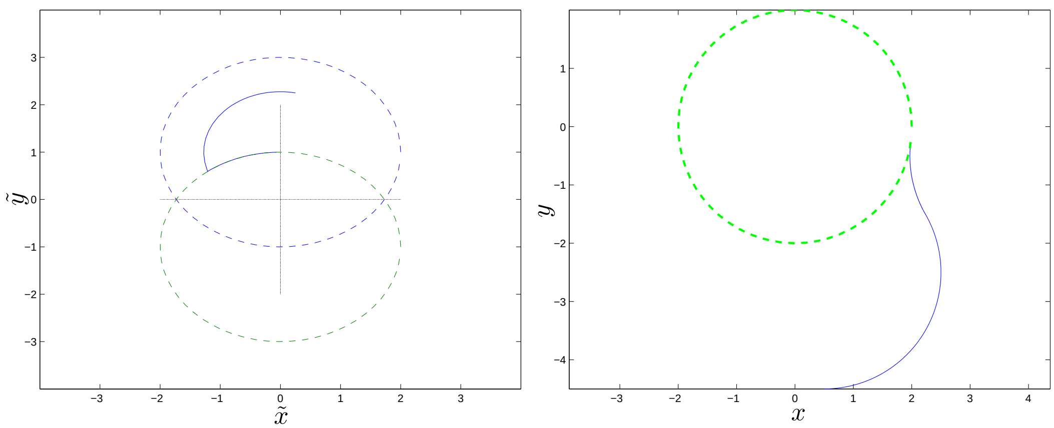


Figure 16: A bang-bang optimal trajectory. A bang-bang solution to problem **(Q)** (left) and the corresponding optimal solution to problem **(P)** (right).

4 Tracking a moving target

The steering methods presented in the previous sections of this paper were concerned with a fixed target. In this section, the results of the previous sections are brought together for the development of a control strategy for tracking a moving target.

This tracking problem has been tackled in different ways. For instance, while some authors use predefined pieces of trajectories called *patterns* that depend on the difference of velocity between the target and the UAV [30, 34] some others prefer to use some simple algorithm based on tangent vector or lateral guidance laws [39, 47] in order to compute in real time the desired path for the UAV. Some authors consider that the tracking problem can be modeled by a predator-prey model which gives waypoints to be interpolated by the UAV [4].

Also, some authors use the same kind of methods as the ones we have described in the previous sections. For instance, in [33] a method using "Lyapunov vector fields" is presented to track a moving target.

Time-optimal problem to manage convoy protection has been considered in [21] where the

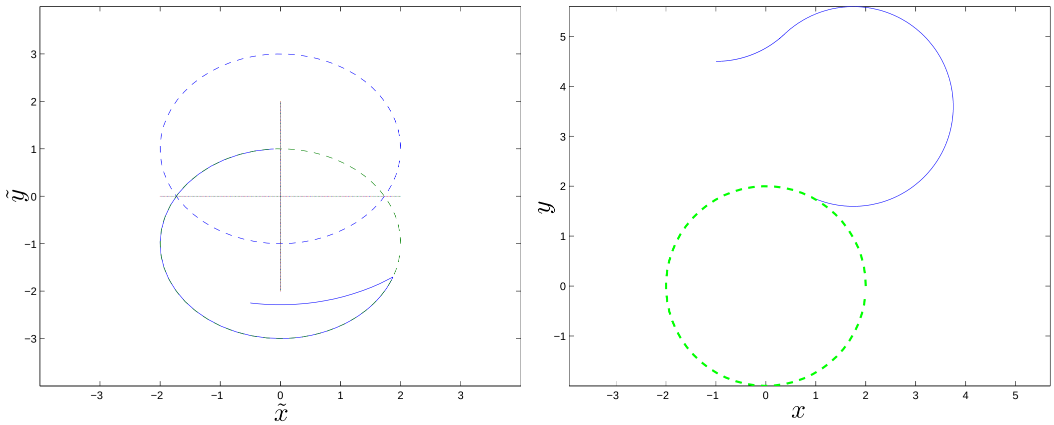


Figure 17: Another bang-bang optimal trajectory. A bang-bang solution to problem **(Q)** (left) and the corresponding optimal solution to problem **(P)** (right).

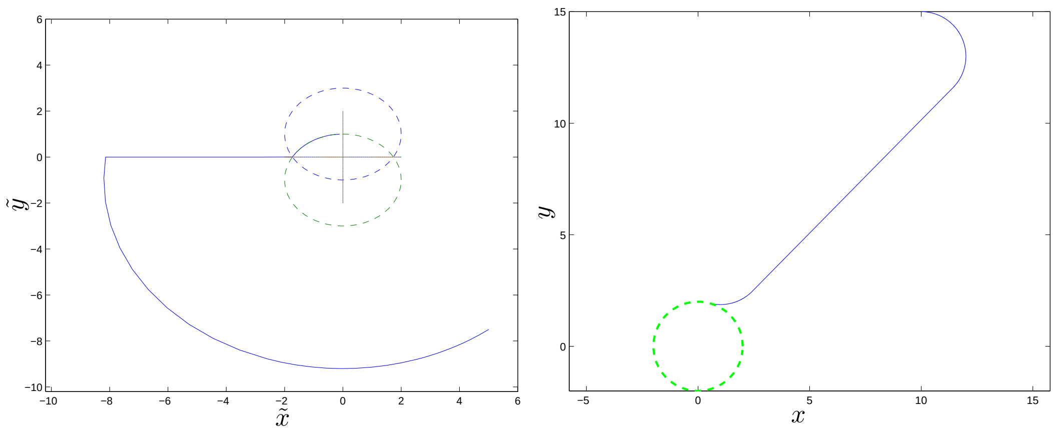


Figure 18: A bang-singular-bang optimal trajectory. A bang-singular-bang solution to problem **(Q)** (left) and the corresponding optimal solution to problem **(P)** (right).

authors consider that the minimum turning radius of UAV is larger than the radius of the protection circle around the target, i.e., the UAV cannot follow this circular trajectory.

4.1 Statement of the problem

Our goal is to make the UAV reach and track a time-parameterized circular final manifold $\mathcal{C}_t$ lying in an horizontal plane above the moving target. Both cases when the future trajectory of the target is known or not are considered.

Let $z = (z_1, z_2, z_3)$ represent the target coordinate in $\mathbb{R}^3$. Define $\mathcal{C}_t$ to be the time-parameterized counterclockwise oriented circle with minimal turning radius $r = 1/u_{\max}$ centered at $(z_1(t), z_2(t))$ in a horizontal plane of a given altitude above the target.

The simple algorithm presented below depicts an open-loop controller which can be used in a high-level controller that is itself used to find waypoints to be sent to the low-level controller performing the closed-loop control of the UAV.

Here for every position of a HALE or MALE UAV, modeled by (2.1) or (2.5) respectively, we would like to find a trajectory steering the considered UAV to the minimum curvature horizontal circle centered at the centroid of a moving target, at a given altitude w.r.t. the target.

It is necessary to make some assumptions on the target behavior. First, we assume that the target speed v_{target} is smaller than the UAV speed v_{uav}, since it is clear that without this assumption the UAV cannot follow the target.

Summing up we consider the following problem

Problem T *Given two initial conditions $z(0) = z_0$ and $q(0) = q_0$ in the corresponding state spaces of the target and the considered UAV, find a pair trajectory-control joining q_0 to $\mathcal{C}_t$ which is admissible for the considered control system.*

4.2 Description of the algorithm

We assume that the past behaviors of both the UAV and the target are known. In practice, the target trajectory is obtained via an external module (as for example a vision based module see [23]).

The following rough algorithm describes a high-level open-loop target tracking controller. We construct the path in the following way

- **STEP 0.** Acquisition of initial data $z_0 \in \mathbb{R}^3$ and $q_0 \in \mathbb{R}^3 \times SO(3)$, the position of the target and UAV respectively.

  Set $t = 0$.

- **STEP 1.** Compute $t^* = \min\{s \in \mathbb{R}_+ \mid q(t+s) \in \mathcal{C}_t)\}$.

- **STEP 2.** Predict the value of $z(t + t^*)$.

- **STEP 3.** Compute the pair trajectory-control $(q(\cdot), u(\cdot))$ to reach the $\mathcal{C}_{t+t^*}$ and apply the control to the UAV.

- **STEP 4.** Set $t = t + \Delta t$ and go to STEP 1, where Δt is some time step, that may either be constant or updated in terms of external informations.

Remark 4.1. There is of course no convergence result for this strategy. Here, we just give an idea on how to apply our previous steering methods in the case of a moving target. Of course the positioning error occurring at **STEP 3** of the previous algorithm should be studied.

Remark 4.2 (more precisions on the different steps of the algorithm)**.**

-At **STEP 1** we compute the final time t^* to reach the $\mathcal{C}_t$. Depending on the method, the value of this final time will be obtained exactly or not. The time-optimal method yields the exact time to reach the final manifold $\mathcal{C}_t$. In this case, t^* is solution to problem **(P)** (or **(Q)**, see Section 3). When the Lyapunov method is used, $\mathcal{C}_t$ is reached asymptotically and the time t^* is the first positive time at which the UAV enters a neiborhood $\mathcal{C}_t^\varepsilon = B(z(t), r+\varepsilon) \bigcap B(z(t), r-\varepsilon)$ of $\mathcal{C}_t$. The parameters ε can then be tuned in order to improve on the performance of the path following algorithm.

-At **STEP 2** we make a prediction of the future position (in space) $z(t + t^*)$ of the target. Of course this position may be obtained in different ways. The method we have used consists of using an observer in order to reconstruct the target velocities. In this work we implemented the so-called extended Kalman filter (EKF for short) (see e.g. [6, 42, 38]) which is well-known for its robustness w.r.t. the noise.

For this purpose, let us assume that the target dynamics is also modeled by an extended Dubins system (see [18])

$$\begin{cases} \dot{z}_1 = u_1 \cos\theta \\ \dot{z}_2 = u_1 \sin\theta \\ \dot{z}_3 = u_2 \\ \dot{\theta} = u_3, \end{cases} \tag{4.1}$$

where the controls u_1, u_2 and u_3 are the target horizontal linear velocity, altitude and yaw angular velocity respectively. The available measurements are the cartesian target coordinate z_1, z_2, z_3, thus, according to (4.1), u_1, u_2, u_3 are clearly observable (we used the local model $\dot{u}_i = 0$, $i = 1, 2, 3$). Note that system (4.1) reduces to the Dubins system when the target moves in a horizontal plane.

4.3 Numerical results

We now present numerical simulations illustrating the tracking algorithm presented in the previous section.

In each case we consider a UAV flying at constant unitary speed and modeled by System (2.1) or System (2.5) depending on the type of UAV we consider (HALE or MALE). We numerically integrate the system during 500 units of time.

For each of the three methods we show simulations for both cases where the target path is known or not. If this information is unknown we use an extended Kalman filter to predict the future target position. Note that the target speed is not assumed to be constant.

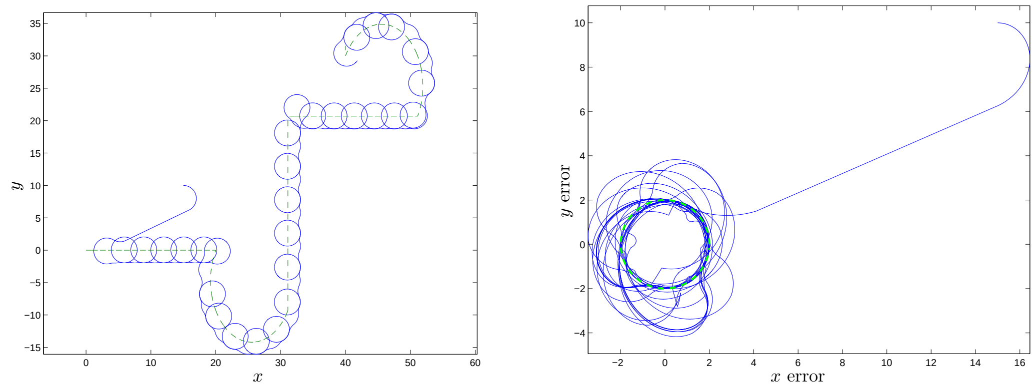
On each figure, the dashed circle represents the trajectory to be followed. The results are drawn in the inertial frame (on the left) and in an orthonormal moving frame attached to the target (on the right) in order to observe the positioning error.

Figures 19 and 20 illustrate the time-optimal tracking method for the same UAV starting from the same initial position. The target that we want to track starts at $(z_1, z_2) = (0, 0)$; its behavior is known on Figure 19 and unknown on Figure 20.

The second couple of figures, Figures 21 and 22, illustrate the Lyapunov tracking method for the same UAV starting from the same initial position. The target that we want to track starts at $(\hat{x}_0, \hat{y}_0) = (0, 0)$; its behavior is known on Figure 21 and unknown on Figure 22.

Figure 23 and Figures 24, illustrate the 3D-Lyapunov tracking method for the same MALE UAV starts horizontally from the point $(10, -20, 50)$ with a yaw direction of 0 radian and that has the curvature constraints $u_{1\,\max} = u_{2\,\max} = 0.1$, $u_{3\,\max} = 0.5$, $a_3 = 0.1$. The values of the parameters ε and σ (see Section 2.2) are 1.2 and 1 respectively. The target that we want to track starts at $(0, 0, 0)$ and is supposed to move in an horizontal plane; its behavior is known on Figure 23 and unknown on Figure 24. In each simulation, $\mathcal{C}_t$ lies in the horizontal plane located at an altitude of 10 units over the target.

Finally, Figure 25 shows a tracking trajectory obtained for a MALE UAV which follows a target which is not moving in a plane. Here the drone must be positioned at 10 units above the target.



(a) (b)

Figure 19: Time-optimal based tracking method in 2D. UAV's trajectory is continuous, target's trajectory is dashed and the $\mathcal{C}_t$ is dashed. The behavior of the target is known. $(x_0, y_0, \theta_0) = (15, 10, 0)$, $(z_1, z_2) = (0, 0)$, and $u_{\max} = -a = 0.5$. (a) Trajectories drawn in the (x, y)-coordinates plane. (b) Trajectories drawn in a (orthonormal) frame attached to the center of $\mathcal{C}_t$.

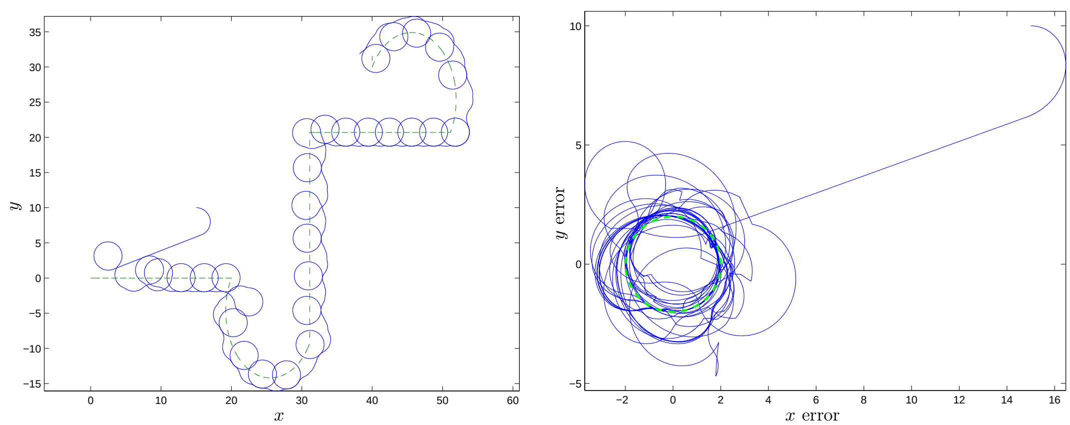


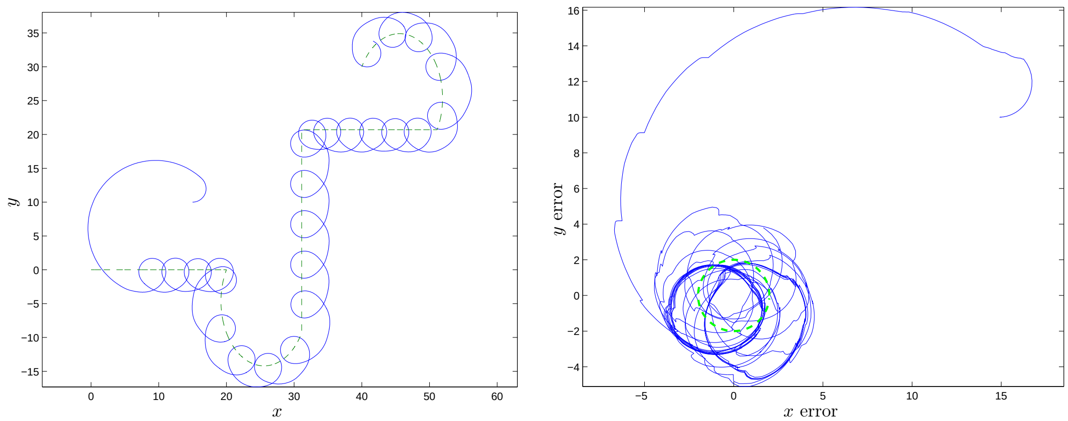
Figure 20: Time-optimal based tracking method in 2D. The only difference with Figure 19 is that, on this figure, the behavior of the target is unknown.

References

[1] A. A. Agrachev and Y. L. Sachkov. *Control theory from the geometric viewpoint*, volume 87 of *Encyclopaedia of Mathematical Sciences*. Springer-Verlag, Berlin, 2004. Control Theory and Optimization, II.

[2] A. A. Agrachev and M. Sigalotti. On the local structure of optimal trajectories in $\mathbf{R}^3$. *SIAM J. Control Optim.*, 42(2):513–531 (electronic), 2003.

[3] A. Ajami, T. Maillot, N. Boizot, J.-F. Balmat, and J.-P. Gauthier. Simulation of a uav ground control station. In *Proceedings of the 9th International Conference of Modeling and Simulation, MOSIM'12*, 2012, *To appear.* Bordeaux, France, 6-8 June, 2012.



(a) (b)

Figure 21: Lyapunov-LaSalle-based tracking method in 2D. UAV's trajectory is continuous, target's trajectory is dashed and $\mathcal{C}_t$ is dashed. The behavior of the target is known. $(x_0, y_0, \theta_0) = (15, 10, 0)$, $(z_1, z_2) = (0, 0)$, and $u_{\max} = 0.5$, $a = 0.1$. (a) Trajectories drawn in the (x, y)-coordinates plane. (b) Trajectories drawn in a (orthonormal) frame attached to the center of $\mathcal{C}_t$
.

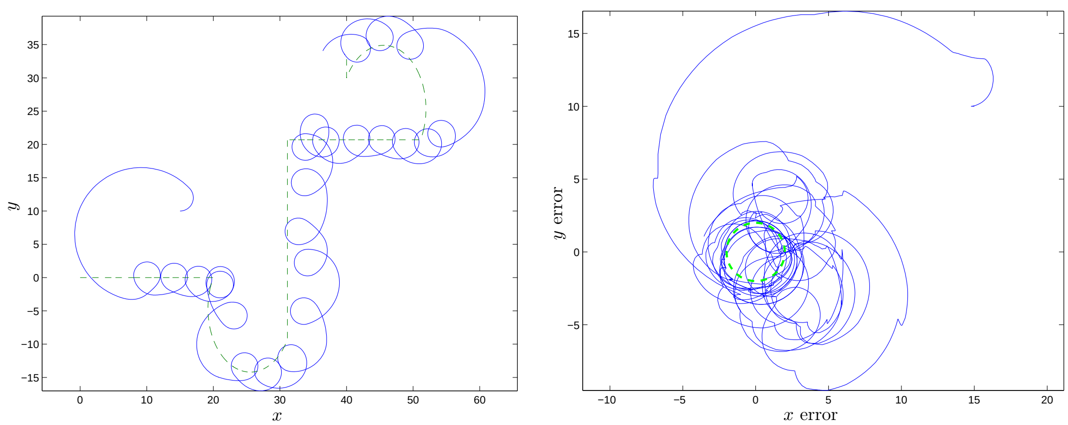


Figure 22: Lyapunov based tracking method in 2D. The only difference with Figure 21 is that, on this figure, the behavior of the target is unknown.

[4] K. Ariyur and K. Fregene. Autonomous tracking of a ground vehicle by a uav. In *American Control Conference, 2008*, pages 669 –671, june 2008.

[5] L. Bertuccelli, A. Wu, and J. How. Robust adaptive markov decision processes: Planning with model uncertainty. *Control Systems, IEEE*, 32(5):96–109, 2012.

[6] G. Besancon. *Nonlinear Observers and Applications.* Lecture notes in control and information sciences. Springer London, Limited, 2007.

[7] A. Bhatia, M. Graziano, S. Karaman, R. Naldi, and E. Frazzoli. Dubins trajectory tracking using commercial off-the-shelf autopilots. In *AIAA Guidance, Navigation, and Control*

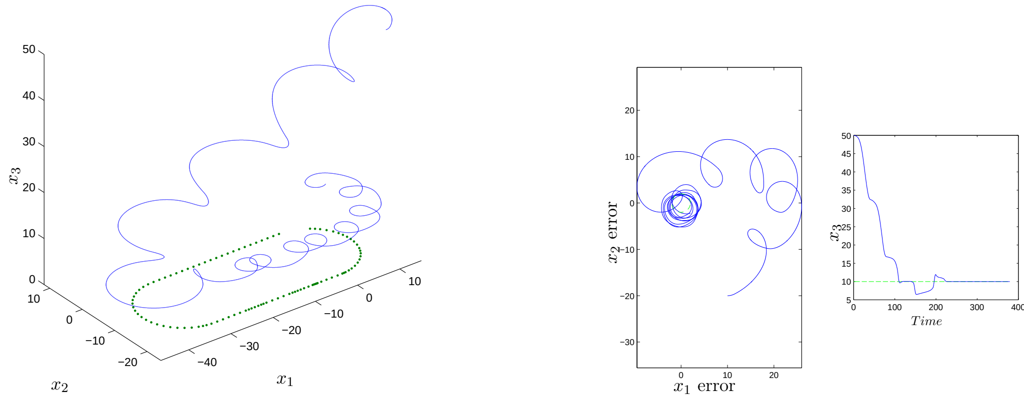


Figure 23: Lyapunov-LaSalle-based tracking method in 3D. The behavior of the target is known.

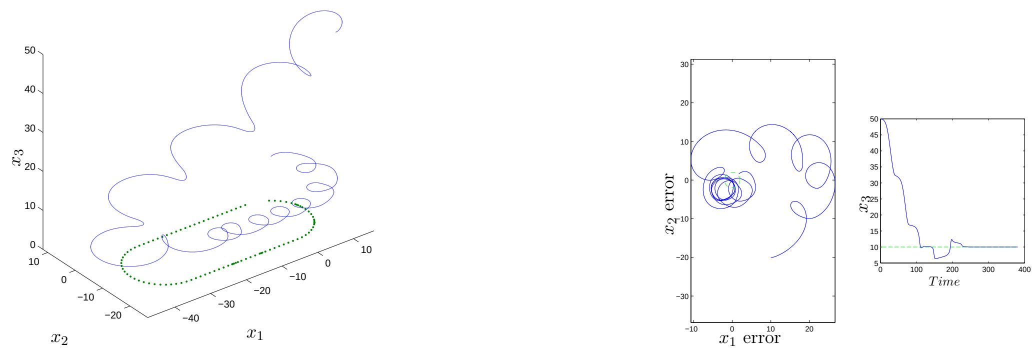


Figure 24: Lyapunov-LaSalle-based tracking method in 3D. The behavior of the target is unknown.

Conference, Honolulu, Hawaii, August 2008.

[8] N. Boizot and J. Gauthier. Motion planning for kinematic systems. *Automatic Control, IEEE Transactions on*, 58(6):1430–1442, 2013.

[9] B. Bonnard, V. Jurdjevic, I. Kupka, and G. Sallet. Transitivity of families of invariant vector fields on the semidirect products of Lie groups. *Trans. Amer. Math. Soc.*, 271(2):525–535, 1982.

[10] U. Boscain and Y. Chitour. Time-optimal synthesis for left-invariant control systems on SO(3). *SIAM J. Control Optim.*, 44(1):111–139 (electronic), 2005.

[11] U. Boscain and B. Piccoli. Extremal synthesis for generic planar systems. *J. Dynam. Control Systems*, 7(2):209–258, 2001.

[12] U. Boscain and B. Piccoli. *Optimal syntheses for control systems on 2-D manifolds*, volume 43 of *Mathématiques & Applications (Berlin) [Mathematics & Applications]*. Springer-Verlag, Berlin, 2004.

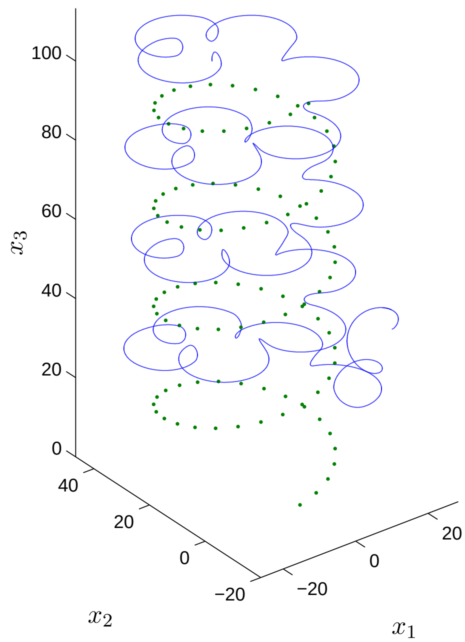


Figure 25: Lyapunov-LaSalle-based tracking method in 3D. The behavior of the target is unknown.

[13] A. Bressan and B. Piccoli. A generic classification of time-optimal planar stabilizing feedbacks. *SIAM J. Control Optim.*, 36(1):12–32 (electronic), 1998.

[14] F. Bullo and A. D. Lewis. *Geometric control of mechanical systems*, volume 49 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 2005. Modeling, analysis, and design for simple mechanical control systems.

[15] D. Campolo, L. Schenato, E. Guglielmelli, and S. Sastry. A lyapunov-based approach for the control of biomimetic robotic systems with periodic forcing inputs. In *Proceedings of 16th IFAC World Congress on Automatic Control (IFAC05)*, 2005.

[16] H. Chen, K. Chang, and C. S. Agate. Tracking with uav using tangent-plus-lyapunov vector field guidance. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 363 –372, july 2009.

[17] Y. Chitour and M. Sigalotti. Dubins' problem on surfaces. I. Nonnegative curvature. *J. Geom. Anal.*, 15(4):565–587, 2005.

[18] H. Chitsaz and S. LaValle. Time-optimal paths for a dubins airplane. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2379 –2384, dec. 2007.

[19] V. Cichella, I. Kaminer, V. Dobrokhodov, E. Xargay, N. Hovakimyan, and A. Pascoal. Geometric 3d path-following control for a fixed-wing uav on so(3). In *AIAA Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics, 2013/05/02 2011.

[20] D. V. Dimarogonas, S. G. Loizou, K. J. Kyriakopoulos, and M. M. Zavlanos. A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. *Automatica*, 42(2):229 – 243, 2006.

[21] X. Ding, A. Rahmani, and M. Egerstedt. Multi-uav convoy protection: an optimal approach to path planning and coordination. *Robotics, IEEE Transactions on*, 26(2):256–268, 2010.

[22] E. Frew, D. Lawrence, C. Dixon, J. Elston, and W. Pisano. Lyapunov guidance vector fields for unmanned aircraft applications. In *American Control Conference, 2007. ACC '07*, pages 371 –376, july 2007.

[23] E. Frew, T. McGee, Z. Kim, X. Xiao, S. Jackson, M. Morimoto, S. Rathinam, J. Padial, and R. Sengupta. Vision-based road-following using a small autonomous aircraft. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 5, pages 3006 – 3015 Vol.5, march 2004.

[24] E. W. Frew, D. A. Lawrence, and S. Morris. Coordinated standoff tracking of moving targets using lyapunov guidance vector fields. *Journal of Guidance, Control, and Dynamics*, 31(2):290–306, 2013/02/21 2008.

[25] J.-P. Gauthier and I. Kupka. *Deterministic observation theory and applications.* Cambridge University Press, Cambridge, 2001.

[26] J.-P. Gauthier and V. Zakalyukin. On the one-step-bracket-generating motion planning problem. *J. Dyn. Control Syst.*, 11(2):215–235, 2005.

[27] C. Goerzen, Z. Kong, and B. Mettler. A survey of motion planning algorithms from the perspective of autonomous uav guidance. *Journal of Intelligent and Robotic Systems*, 57(1-4):65–100, 2010.

[28] M.-D. Hua, T. Hamel, P. Morin, and C. Samson. A control approach for thrust-propelled underactuated vehicles and its application to vtol drones. *Automatic Control, IEEE Transactions on*, 54(8):1837 –1853, aug. 2009.

[29] M.-D. Hua, T. Hamel, P. Morin, and C. Samson. A control approach for thrust-propelled underactuated vehicles and its application to VTOL drones. *IEEE Trans. Automat. Control*, 54(8):1837–1853, 2009.

[30] K. L. Kokkeby, R. P. Lutter, M. L. Munoz, F. W. Cathey, J. D. Hilliard, and T. L. Olson. System and methods for autonomous tracking and surveillance, 06 2009.

[31] J. P. LaSalle. Stability theory for ordinary differential equations. *J. Differential Equations*, 4:57–65, 1968.

[32] J.-P. Laumond, editor. *Robot motion planning and control*, volume 229 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag London Ltd., London, 1998. Available online at http://www.laas.fr/∼jpl/book.html.

[33] D. Lawrence, E. Frew, and W. Pisano. Lyapunov vector fields for autonomous uav flight control. In *AIAA Guidance, Navigation and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics, 2013/02/21 2007.

[34] J. Lee, R. Huang, A. Vaughn, X. Xiao, J. K. Hedrick, M. Zennaro, and R. Sengupta. Strategies of path-planning for a uav to track a ground vehicle. *AINS Conference*, 2003, 2003.

[35] S. Park, J. Deyst, and J. P. How. A new nonlinear guidance logic for trajectory tracking. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics, 2004.

[36] B. Piccoli. Classification of generic singularities for the planar time-optimal synthesis. *SIAM J. Control Optim.*, 34(6):1914–1946, 1996.

[37] B. Piccoli and H. J. Sussmann. Regular synthesis and sufficiency conditions for optimality. *SIAM J. Control Optim.*, 39(2):359–410 (electronic), 2000.

[38] C. G. Prévost, A. Desbiens, and D. Gagnon, Ericand Hodouin. Uav optimal cooperative target tracking and collision avoidance of moving objects. In Chung, M. Jin, Misra, and Pradeep, editors, *The International Federation of Automatic Control, Vol. 17*, pages 5724–5729, 2008.

[39] F. Rafi, S. Khan, K. Shafiq, and M. Shah. Autonomous target following by unmanned aerial vehicles. *Unmanned Systems Technology VIII*, 6230(1):623010, 2006.

[40] Y. Sachkov. Control theory on lie groups. *Journal of Mathematical Sciences*, 156:381–439, 2009.

[41] H. Schättler. On the local structure of time-optimal bang-bang trajectories in $\mathbf{R}^3$. *SIAM J. Control Optim.*, 26(1):186–204, 1988.

[42] K. D. Sebesta and N. Boizot. Real-time adaptive high-gain ekf, applied to a quadcopter inertial navigation system. *IEEE Trans. on Indus. Elec.*, To appear.

[43] M. Sigalotti and Y. Chitour. Dubins' problem on surfaces. II. Nonpositive curvature. *SIAM J. Control Optim.*, 45(2):457–482 (electronic), 2006.

[44] P. Souères, A. Balluchi, and A. Bicchi. Optimal feedback control for route tracking with a bounded-curvature vehicle. *Internat. J. Control*, 74(10):1009–1019, 2001.

[45] P. Souères and J.-P. Laumond. Shortest paths synthesis for a car-like robot. *IEEE Trans. Automat. Control*, 41(5):672–688, 1996.

[46] H. J. Sussmann. Regular synthesis for time-optimal control of single-input real analytic systems in the plane. *SIAM J. Control Optim.*, 25(5):1145–1162, 1987.

[47] P. Theodorakopoulos and S. Lacroix. A strategy for tracking a ground target with a uav. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1254 –1259, sept. 2008.

[48] G. Walsh, R. Montgomery, and S. Sastry. Optimal path planning on matrix lie groups. In *Decision and Control, 1994., Proceedings of the 33rd IEEE Conference on*, volume 2, pages 1258 –1263 vol.2, dec 1994.

[49] E. Xargay, V. Dobrokhodov, I. Kaminer, A. Pascoal, N. Hovakimyan, and C. Cao. Time-critical cooperative control of multiple autonomous vehicles: Robust distributed strategies for path-following control and time-coordination over dynamic communications networks. *Control Systems, IEEE*, 32(5):49–73, 2012.

[50] S. Zhu, D. Wang, and C. Low. Ground target tracking using uav with input constraints. *Journal of Intelligent & Robotic Systems*, 69:417–429, 2013.

3.3 Généralisation des méthodes pour atteindre n'importe quel pattern

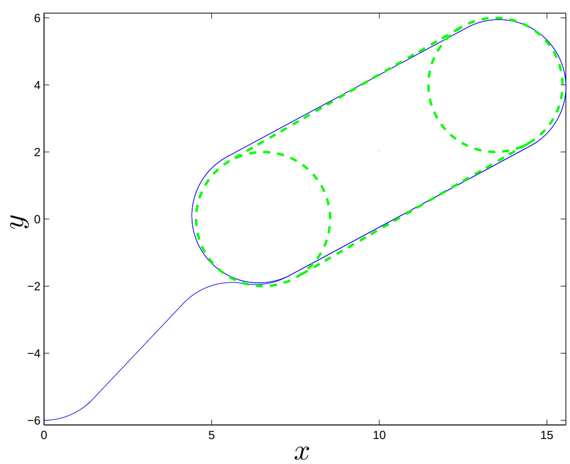
Dans le paragraphe précédent, nous avons proposé des méthodes de planification permettant de rejoindre un cercle.

Dans le but de planifier une trajectoire, pour tous types de missions, il faut que le pattern visé puisse être différent d'un cercle (e.g., un hippodrome ou un huit), cf. paragraphe 1.1.

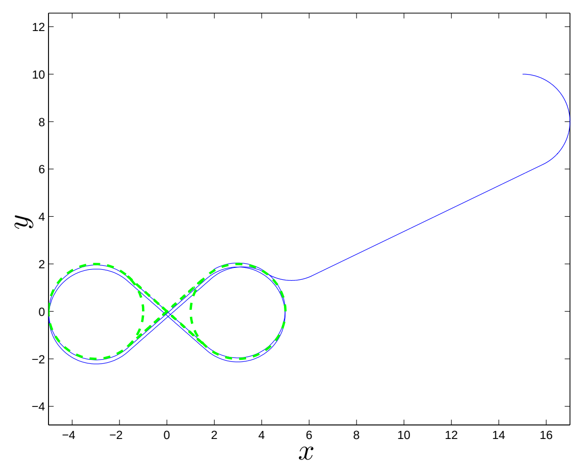
En s'inspirant de [84], pour rejoindre n'importe quel pattern, nous utilisons une des méthodes décrites précédemment en visant un cercle inscrit au pattern considéré.

Nous utilisons deux cercles inscrits pour définir les patterns en forme d'hippodrome et de huit. La position de ces cercles définit l'aspect du pattern (e.g., longueur, orientation).

Un exemple d'application est représenté en Figure 3.8. Pour calculer ces trajectoires, nous avons utilisé la méthode de planification temps-minimal.



(a)



(b)

FIGURE 3.8 – Une trajectoire optimale en temps reliant (a) un hippodrome, (b) un huit.

3.4 Comment l'appareil peut suivre une cible dans un milieu encombré ?

Le but est d'appliquer les résultats exposés précédemment pour effectuer du suivi de convoi en milieu encombré. Cette planification est possible en s'inspirant de :

1. La planification point-point en milieu encombré (cf. paragraphe 2.5.3),
2. La méthode de suivi de convoi en milieu non contraint (cf. paragraphe 3.2).

Les notations sont celles du paragraphe 3.2. La trajectoire est construite de la manière suivante :

Étape 0 Acquisition des données initiales $q_{\text{cible}_0} \in \mathbb{R}^3$ et $q_0 \in \mathbb{R}^3 \times SO(3)$, les positions de la cible et du vecteur respectivement.
Poser $t := 0$.

Étape 1 Calcule de $t^* = \min\{\tau \in \mathbb{R}_+ \mid q(t+\tau) \in \mathscr{C}_t)\}$.

Étape 2 Prédiction de la valeur de $q_{\text{cible}}(t+t^*)$.

Étape 3 Calcule du couple $(q(\cdot), u(\cdot))$ permettant de rejoindre le pattern $\mathscr{C}_{t+t^*}$ en utilisant la méthode suivante :
Étape 3.0 Initialisation du point courant (position du vecteur).
Étape 3.1 Recherche du fil d'Ariane reliant le point courant à $q_{\text{cible}}(t+t^*)$.
Étape 3.2 Recherche d'une trajectoire reliant le point courant et un pattern centré autour d'un point du fil d'Ariane et le plus éloigné possible.
Étape 3.3 Mise à jour du point courant et retour à l'étape 3.1, tant que celui-ci n'est pas dans un voisinage du point d'arrivée.
Étape 3.4 Appliquer le contrôle au vecteur.

Étape 4 Poser $t := t + \Delta t$ et recommencer à partir de l'étape 1. Δt est un pas de temps fixe ou actualisé en fonction d'un module externe.

Remarque 3.6. Le calcul de t^* à l'étape 1 peut être obtenu en appliquant une procédure similaire à celle de l'étape 3.

La méthode de planification point-pattern de l'étape 3.2 est choisie parmi celles présentées au paragraphe 3.2.

Des exemples d'applications sont donnés en Figures 3.9 et 3.10. La trajectoire du convoi suivi est en pointillés. Une méthode de planification point-pattern temps-minimal a été utilisée pour obtenir la trajectoire de la première figure (Fig. 3.9). Nous avons utilisé la méthode basée sur le principe de LaSalle pour calculer la trajectoire de la Figure 3.10.

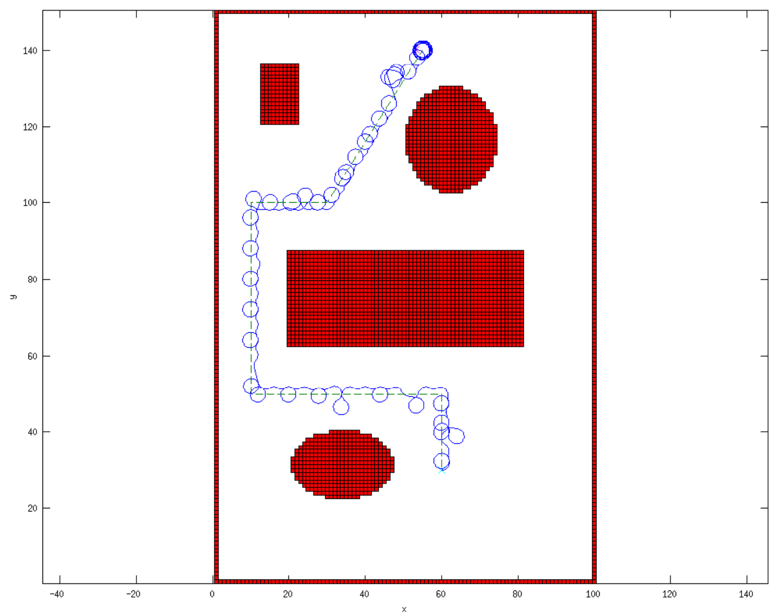


FIGURE 3.9 – Planification de trajectoires en milieu contraint utilisant la méthode temps-minimal. La trajectoire du convoi est en pointillés.

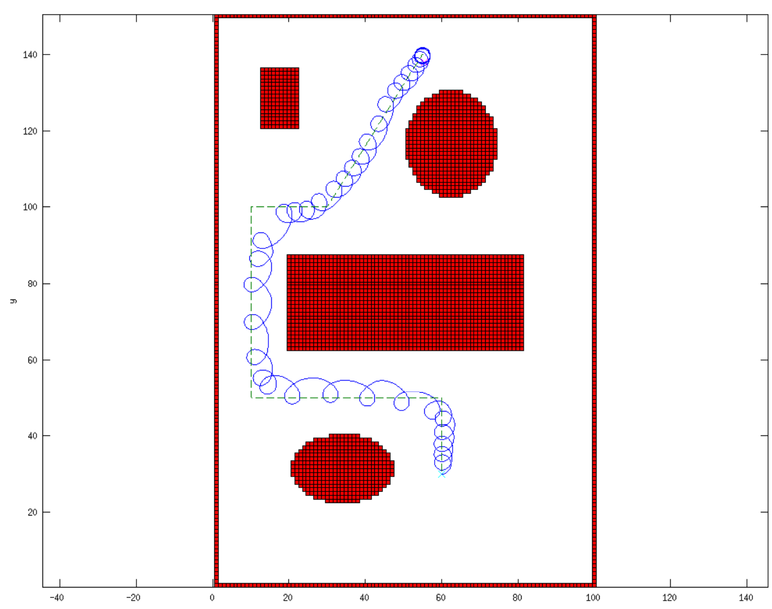


FIGURE 3.10 – Planification de trajectoires en milieu contraint utilisant la méthode basée sur le principe de LaSalle 2D. La trajectoire du convoi est en pointillés.

Chapitre 4

Conclusion

Le travail effectué dans cette première partie concerne la planification de trajectoires pour des drones de type HALE ou MALE, qui peuvent être modélisés par des systèmes différentiels non-linéaires simples.

Le chapitre 2 nous a permis d'exposer les méthodes usuelles de résolution de ce type de problème. Les premiers résultats que nous avons obtenus pour la planification vers un point, ont été étendus pour prendre en compte des contraintes environnementales.

Le chapitre 3 a apporté des réponses au problème de couplage vecteur/capteur. Nous avons proposé une méthode d'asservissement de la trajectoire du couple drone/caméra donnant plus ou moins d'importance à l'un ou à l'autre par le biais de pondérations.

Nous avons remarqué, dans le cas d'une caméra non contrainte, que ce problème de couplage se réduit au problème de planification du vecteur. Dans ce cas, une méthode indépendante de contrôle de la caméra a été proposée.

Le problème de planification a été traité pour une mission impliquant l'utilisation de patterns. Nous avons exposé deux méthodes de planification point-pattern. Ces deux méthodes, basées respectivement sur le principe de LaSalle et sur le PMP, ont été testées numériquement.

Pour finir, nous avons proposé et appliqué un algorithme de suivi de cible, basé sur les méthodes de planification vers un pattern.

Dans toute cette première partie, des algorithmes de planification point-point et point-pattern ont été étudiés. Parmi toutes ces méthodes de calcul de trajectoires, certaines sont obtenues par la minimisation d'une fonction de coût.

Dans la partie suivante, nous proposons une méthode de construction d'une fonction de coût permettant de calculer des trajectoires les plus proches possibles de celles effectuées par les pilotes.

Deuxième partie

Une méthode anthropomorphique : l'utilisation du contrôle optimal inverse

Chapitre 5

Introduction

Dans la partie précédente, nous avons étudié des méthodes de planification point-point et point-pattern. Parmi toutes ces méthodes de calcul de trajectoires, certaines sont obtenues par la minimisation d'une fonction de coût.

Le problème que nous allons étudier, dans cette partie, est le suivant : comment proposer des trajectoires voisines de celles qu'un pilote expérimenté serait amené à décrire s'il était directement aux commandes de l'appareil ?

En d'autres termes, à partir du comportement d'un aéronef, nous souhaitons extraire un coût afin de l'utiliser au sein des modules de planification précédemment présentés.

Cette approche, connue sous le nom de contrôle optimal inverse, a été appliquée à des problèmes de biomécanique humaine (quelques références sont données page 101). Cette méthode d'analyse de données très novatrice est basée sur un paradigme aujourd'hui largement accepté en neurophysiologie selon lequel, parmi tous les mouvements possibles ceux accomplis satisfont un critère d'optimalité [134].

L'utilisation de cette analyse est importante puisqu'elle nous permet d'appliquer les règles de l'Organisation de l'Aviation Civile Internationale, et notamment l'article 2.13 de la circulaire 328 de l'OACI [74].

Article 2.13 de la circulaire 328 de l'OACI

A key factor in safely integrating UAS in non-segregated airspace will be their ability to act and respond as manned aircraft do. [...]

International Civil Aviation Organization

Dans le chapitre 6, nous exposons quelques travaux déjà effectués dans le domaine. Le chapitre 7 présente nos propres résultats qui constituent en fait un article accepté dans la revue *ESAIM : Control, Optimisation and Calculus of Variations* (à paraître en 2013).

Chapitre 6

Méthodes bio-inspirées et problème de contrôle optimal inverse

Sommaire

6.1 Les méthodes bio-inspirées

La bio-inspiration, ou biomimétisme, est une philosophie de développement basée sur l'inspiration par la nature.

De nombreux travaux ont proposé de modéliser et créer des robots en mimant des insectes ou des animaux. C'est le cas par exemple dans [112] où les auteurs s'inspirent du comportement des mouches pour effectuer de la planification sommaire de trajectoires, afin de longer un mur.

D'autres encore s'inspirent des criquets pour créer des capteurs de positionnement appliqués à des véhicules aériens [114].

Certains tentent d'imiter le comportement des jongleurs pour effectuer de la manipulation d'objets avec des bras robotiques [7]. Cette même équipe a aussi travaillé sur un robot mimant une salamandre [65].

Tous ces travaux ont été possibles grâce à des modélisations et des études de comportements naturels. Par exemple, le corps humain a beaucoup été étudié [67, 118, 9, 11]. De plus, grâce à des études comportementales, certains principes naturels ont été découverts, e.g., le principe d'inactivation [20].

Afin de traiter ce problème de biomimétisme, certains auteurs considèrent que la nature peut être analysée via un problème de contrôle optimal inverse [30, 87, 17]. Ce problème est explicité dans le paragraphe suivant.

6.2 Le problème de contrôle optimal inverse

Le problème de contrôle optimal inverse est similaire à de la cryptanalyse. Rappelons d'abord que la cryptanalyse a pour but de retrouver la clef permettant de décoder un texte chiffré, écrit par une tierce personne. Le principe de résolution du problème de contrôle optimal est le même : il permet de trouver le critère d'optimalité (la clef) d'un problème de contrôle optimal, à partir d'observations (le texte à décrypter).

Comme pour effectuer une cryptanalyse, nous avons besoin de poser des hypothèses plausibles sur le phénomène étudié. Ces hypothèses concernent en général la classe de systèmes différentiels choisie pour modéliser le phénomène, ainsi que la classe de régularité de ces systèmes et du critère recherché.

Il faut ensuite retrouver, à partir d'observations, le critère d'optimalité minimisé. Le problème se formule de la manière suivante

Problème de contrôle optimal inverse

Étant donnés un système de contrôle $\dot{q} = f(q,u)$ et un ensemble Γ de trajectoires admissibles observées, trouver un critère $C(q(\cdot),u(\cdot))$ tel que toute trajectoire $\gamma \in \Gamma$ soit solution de

$$\inf\{C(q(\cdot),u(\cdot)) \mid \dot{q} = f(q,u), \quad q(0) = \gamma(0), \quad q(T) = \gamma(T)\}$$

En acceptant le paradigme de neurophysiologie selon lequel, parmi tous les mouvements possibles ceux accomplis satisfont un critère d'optimalité [134], les actions humaines sont analysables via un problème de contrôle optimal inverse.

Voici une liste, non exhaustive, des phénomènes étudiés :

- Mouvement de pointage du bras (B. Berret, J.-P. Gauthier, F. Jean, E. Todorov, F. Jean) [20, 99, 19, 99]
- Mouvement de saccade des yeux (F. Jean)
- Locomotion humaine (Y. Chitour, F. Chittaro, F. Jean, J.-P. Laumond, P. Mason, E. Todorov) [10, 9, 11, 39, 52, 17, 41]

Une des difficultés, dans ces travaux, est due à l'aspect expérimental de la résolution : la présence de bruits de mesure ne facilite pas la reconstruction du coût. Une autre difficulté est la modélisation du problème : il n'existe pas toujours de modèle permettant d'expliquer la dynamique d'un système biologique. À cause de ces difficultés, la précision du critère reconstruit peut être limitée. Néanmoins, des travaux récents, dont celui de F. Jean *et al.* [41], donnent des résultats de robustesse du problème. Ces résultats permettent de savoir si ces incertitudes ont une grande influence sur la reconstruction du critère.

Dans le chapitre suivant, nous modélisons et étudions le problème de contrôle optimal inverse pour l'appliquer aux trajectoires de pilotes expérimentés.

Chapitre 7

Une méthode d'estimation du coût optimisé par des pilotes humains

L'article présenté dans ce Chapitre a été publié dans le journal *ESAIM : Control, Optimisation and Calculus of Variations* [5].

ESAIM: COCV
DOI: 10.1051/cocv/2012043

ESAIM: Control, Optimisation and Calculus of Variations
www.esaim-cocv.org

HOW HUMANS FLY

Alain Ajami[1], Jean-Paul Gauthier[1,2], Thibault Maillot[1] and Ulysse Serres[3,4]

Abstract. This paper is devoted to the general problem of reconstructing the cost from the observation of trajectories, in a problem of optimal control. It is motivated by the following applied problem, concerning HALE drones: one would like them to decide by themselves for their trajectories, and to behave at least as a good human pilot. This applied question is very similar to the problem of determining what is minimized in human locomotion. These starting points are the reasons for the particular classes of control systems and of costs under consideration. To summarize, our conclusion is that in general, inside these classes, three experiments visiting the same values of the control are needed to reconstruct the cost, and two experiments are in general not enough. The method is constructive.

The proof of these results is mostly based upon the Thom's transversality theory.

This study is partly supported by FUI AAP9 project SHARE, and by ANR Project GCM, program "blanche", project number NT09-504490.

Mathematics Subject Classification. 93C10, 49J15, 93C41, 93C15, 34K35.

Received May 14, 2012. Revised October 16, 2012.
Published online June 3, 2013.

1. Introduction

This study holds in the context of the french FUI SHARE project (see [2]), and authors are granted from the project.

From the kinematic point of view, a rough HALE drone (high altitude, long endurance drone) is governed by the standard Dubins equations:

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{\theta} = u. \end{cases} \tag{1.1}$$

These equations express that the drone moves on a perfect plane (perfect constant altitude), at perfect constant speed 1, moves in the direction of its velocity vector, and is able to turn right and left.

Keywords and phrases. Inverse optimal control, anthropomorphic control, transversality.

[1] Université du Sud-Toulon-Var, LSIS, UMR CNRS 7296, B.P 20132, 83957 La Garde Cedex, France. `alain.ajami@univ-tln.fr`; `jean-paul.gauthier@univ-tln.fr`; `thibault.maillot@univ-tln.fr`

[2] INRIA GECO Project

[3] Université de Lyon, 69 622 Lyon, France

[4] Université Lyon 1, Villeurbanne; LAGEP, UMR CNRS 5007, 43 bd du 11 novembre 1918, 69100 Villeurbanne, France. `ulysse.serres@univ-lyon1.fr`

These equations were also considered by people studying human locomotion: see the seminal works of Chitour-Jean-Mason [8], and Chittaro-Jean-Mason [10]. See also older but basic works of Laumond [3–5], who originally introduced this point of view.

Another typical result related with cost reconstruction in human movements is [6, 7, 11]. A part of the methodology in [6, 7, 11] is very similar to what we do here in.

Note that the rough model (1.1) is more pertinent as a model of the kinematics of HALE drones than as a model of human locomotion: actually, for HALE drones, the velocity is almost really constant[5].

Once this assumption of constant velocity is accepted, the natural requirement of invariance under motions of the plane leads to these equations.

In both cases but for different reasons, the assumption is that some integral cost is minimized, to connect **in free time**, the initial and target point. Again, the natural requirement of invariance with respect to motions of the plane leads to an integral cost of the form:

$$C(u(\cdot)) = \int_0^T L(u(t), \dot{u}(t), \ldots, u^{(k)}(t))\mathrm{d}t \tag{1.2}$$

In the case of human locomotion, a very interesting argument of dimension of a certain set shows that $k = 1$ in equation (1.2) above (see [13]). We discuss this in more details in Sections 3.3, 5. Shortly, the results here provide a clear method to validate this assumption $k = 1$.

It turns out, and this is quickly shown in Section 5, that the measurement of two different trajectories is enough to reconstruct the cost $L(u)$ (over the set of values of $u(t)$ visited during the two experiments), provided that both experiments visit the same value of the control (at different times, maybe).

These considerations are the starting point of this study, and the reason to consider the following class of nonlinear control affine systems, with single control u, and of costs $C(u(\cdot))$:

$$\begin{cases} \dot{x} = f(x, y) \\ \dot{y} = u \end{cases} \tag{1.3}$$

where x is n-dimensional, and the cost

$$C(u(\cdot)) = \int_0^T L(u(t))\mathrm{d}t. \tag{1.4}$$

Surprisingly, our main result is the following rough statement, the precise result being Theorem 4.1, Section 4.1.

Theorem 1.1 (rough statement)**.** To reconstruct the cost (1.4), three experiments are in general enough provided that they visit the same value of the control, at some (maybe different) instants. Two experiments are in general not enough.

Hence, in fact, the Dubins case is, inside its class, very particular. This is due to the symmetries (invariance of the problem under the action of motions of the plane).

Moreover, in both cases (Dubins case and case (1.3)), **the reconstruction of L can be made only up to a linear term, which is in fact totally irrelevant**: adding a linear term to L does not change the set of optimal trajectories. On the contrary, for completely general affine control systems, this additional degree of freedom does not (generically) exist.

In the paper, we finish by considering (more quickly) the case of a general control affine system, $\dot{q} = F_0(q) + uF_1(q)$ (Sect. 6). It turns out that the main Theorem 1.1 still holds. Even a bit more: now $L(u)$ is completely determined by three experiments (not up to some linear term).

[5]The argument of possible length reparametrization used in [8, Sect. 2, item (ii), p. 150], seems questionable.

The organization of the paper is as follows:

- In Sections 2, 3, we give a clear statement of the problem, together with all prerequisites necessary to the mathematical precise understanding of the paper. A key lemma for the proofs (Lem. 3.17) is established. The very short Section 3.3 concerns our conclusion about the validation of the assumption that the cost $C(u(\cdot))$ is of the form (1.4).
- Section 4 is devoted to the results of our study. In Section 4.1 we give a clear statement of our main result and in Section 4.2 we give the crucial preliminaries for the proof of the main result (Thm. 4.1). Section 4.3.2 regroups, besides the final proof, all technical lemmas, and the proof of the fact that in general two experiments are not enough.
- Section 5 deals with Dubins case, and the application to HALE drones.
- Section 6 cares about the case of general control affine systems (very shortly).

2. Statement of the problem

In this paper, smooth means C^∞. Also, we do not distinguish between row and column vectors, the distinction being clear from the context.

2.1. Systems under consideration

As we said in the introduction, we deal with single input nonlinear control affine systems of the form (1.3), with $(x, y) \in X \times Y$, the state variable and $u \in \mathbb{R}$, the control variable. Here, X (resp. Y) is an n-dimensional (resp. one-dimensional) connected, Hausdorff and paracompact smooth manifold.

Setting $q = (x, y)$ and $Q = X \times Y$, we rewrite the system (1.3) as

$$\dot{q} = F_0(q) + uF_1(q), \qquad q \in Q, \quad u \in \mathbb{R}, \tag{2.1}$$

where $F_0 = (f, 0)$ and $F_1 = (0, 1)$ are smooth vector fields on Q. Depending on the context, we will use freely the most convenient notation between both.

Let $\mathcal{F}$ denote the set of smooth control systems form (1.3) or (2.1), *i.e.*, elements of $\mathcal{F}$ are smooth vector fields F_0 on Q such that $F_0 : Q \to TX \times \{0_{TY}\}$. Equivalently, elements of $\mathcal{F}$ are y-parametrized families of smooth vector fields f over X, *i.e.*, smooth sections of a vector bundle over Q with fiber T_xX. Let $J^k\mathcal{F}$ denote the bundle of k-jets of systems in $\mathcal{F}$.

The set of admissible control functions $u(\cdot)$ is as usual $\mathbb{L}^\infty_{\text{loc}}(\mathbb{R}_+, \mathbb{R})$, and the set of admissible trajectories is the set of solutions of system (1.3) or (2.1) corresponding to an admissible control function $u(\cdot)$. The domain of our control functions will be a certain time interval $[t_1, t_2]$ depending on $u(\cdot)$. If necessary, it is always possible to assume that $t_1 = 0$.

2.2. Problem under consideration

2.2.1. Main assumption

Our main assumption in the paper is: all trajectories occuring in the observations are solutions of an optimal control problem of the following form, that we shall denote by $(\mathbf{P}_L)$.

$(\mathbf{P}_L)$ Minimize the integral cost (1.4) among all admissible controls $u(\cdot)$ steering system (2.1) from a source point q_0 to a target point q_1 in free final time T.

We do not consider arbitrary functions L in the functionals $C(u(\cdot))$ above: indeed, we choose a class $\mathcal{L}$ ensuring that the optimal control problem $(\mathbf{P}_L)$ has a solution for each L in $\mathcal{L}$. Following [8,10], $\mathcal{L}$ is the set of $L : \mathbb{R} \to \mathbb{R}$ meeting the following assumptions:

(A1) $L(\cdot)$ is strictly positive and smooth;
(A2) $L(\cdot)$ is strictly convex with $L''(u) > 0$ (*i.e.*, strictly strongly convex);
(A3) $\lim_{|u|\to+\infty} L(u)/|u| = +\infty$.

2.2.2. The different notions of experiments

Among the set of admissible pairs $(q(\cdot), u(\cdot))$, we shall distinguish the ones called *experiments:*

Definition 2.1 (experiment)**.** An admissible trajectory $(q(\cdot), u(\cdot))$ is called an *experiment* if there exists $L \in \mathcal{L}$ such that $(q(\cdot), u(\cdot))$ solves $(\mathbf{P}_L)$.

Remark 2.2. Since a solution of system (2.1) is uniquely determined by the initial condition q_0 in Q and the admissible control function $u(\cdot)$, we may identify an experiment $(q(\cdot), u(\cdot))$ with a pair $(q_0, u(\cdot))$. For simplicity, we shall denote indifferently both by γ.

We also denote by $\mathit{\Gamma}_f$ a family of experiments whose corresponding trajectories $q(\cdot)$ are solutions to system (2.1).

Definition 2.3 (compatible experiments)**.** A family $\mathit{\Gamma}_f$ of admissible pairs $\gamma = (q, u(\cdot))$ is a family of compatible experiments if there exists (a common) $L \in \mathcal{L}$ such that every $\gamma \in \mathit{\Gamma}_f$, solves $(\mathbf{P}_L)$.

Definition 2.4 (monotonic experiment)**.** An experiment $(q(\cdot), u(\cdot))$ is monotonic if the control function $u(\cdot)$ is smooth monotonic (strictly, *i.e.*, $u'(t)$ does not vanish for any t).

From now on we consider monotonic experiments only. In particular, the mapping $u \mapsto t(u)$ is well defined, continuous and smooth, *i.e.*, belongs to $\mathcal{U} = \mathcal{C}^\infty(\mathbb{R}_+, \mathbb{R})$. The smoothness assumption, that may look not very reasonable, will be justified later (Lem. 3.5).

In the following, with a little abuse of notations, a time dependent function g is written $g(u)$ instead of $g(t(u))$ whenever necessary. Also, we use the dot $\dot{}$ to denote the derivative with respect to time t, and the prime $'$ to denote the derivative with respect to variable u.

Definition 2.5 (different experiments)**.** Several experiments $(q^i(\cdot), u^i(\cdot))$, $i = 1, \ldots, \ell$, that visit common control values (*i.e.*, such that the ranges of the associated control functions $u^i(\cdot)$ overlap) are said to be different if there are points t such that $q^i(t) \neq q^j(t)$ for all $i \neq j$, on the overlapping range of control values.

Remark 2.6. A few obvious but useful observations are in order:

(i) The control of a monotonic experiment can take the value zero at most once.
(ii) Let $\mathit{\Gamma}_f$ be a family of experiments that visit common control values, namely,

$$\exists\, u_0 \in \mathbb{R} \quad \forall\, \gamma = (q_0, u(\cdot)) \in \mathit{\Gamma}_f \quad \exists\, t(\gamma) \in \mathbb{R}_+ \quad | \quad u(t(\gamma)) = u_0,$$

we may assume without loss of generality that $t(\gamma) = 0$ for each experiment (shift of time on the experiments).

2.2.3. Statement of the inverse optimal control problem

The inverse optimal control problem that we shall denote by $(\reflectbox{$\mathbf{P}_L$})$ is the following.
$(\reflectbox{$\mathbf{P}_L$})$ Given a family $\mathit{\Gamma}_f$ of experiments, find a cost L in $\mathcal{L}$ such that every $\gamma \in \mathit{\Gamma}_f$ solves $(\mathbf{P}_L)$.

3. Study of the optimal control problem and its inverse

3.1. Study of the optimal control problem $(\mathbf{P}_L)$

3.1.1. Existence of optimal controls

We follow exactly the same lines as in [10, Sect. 3.1] for questions dealing with the existence of optimal controls. The growth condition $L(u) \geqslant c|u|^p$ with $c > 0$ and $p > 1$ used in [10, Sect. 2.1], has been weakened to **A3** according to [18, Thm. 2.8.1], and thus we can state the following.

Theorem 3.1. For every $q_0,\ q_1 \in Q$ such that q_1 is reachable from q_0, there exists a real positive time T^* and a control function $u(\cdot) \in \mathbb{L}^1([0, T^*])$ such that the corresponding trajectory $q(\cdot)$ solves problem $(\mathbf{P}_L)$.

Notice that Theorem 3.1 does not provide the boundedness of the optimal controls. It will follow from Lemma 3.5.

Remark 3.2. Observe that the existence of a bounded positive T^* is not completely trivial and follows from the special form of the cost function L. Indeed, it follows easily from assumptions **A1** and **A3** that there exists a positive real number α such that $L(u) > \alpha$ for every u. Now, if q_1 is attainable from q_0, let $u_n(\cdot) : [0, T_n] \to Q$ be a minimizing sequence. Then,

$$\alpha T_n < C(u_n(\cdot)) < +\infty,$$

which implies that T_n must be uniformly bounded.

3.1.2. Application of the Pontryagin Maximum Principle

Since we do not know yet that optimal controls are bounded in the $\mathbb{L}^\infty$ topology, we cannot apply the PMP in its classical version. Nonetheless, it is easy to check that problem $(\mathbf{P}_L)$ meets all the hypotheses required in [18, Thm. 8.7.1] (a more refined version of PMP valid for unbounded controls), and thus we can state the following.

Proposition 3.3. Every solution to $(\mathbf{P}_L)$ satisfies the PMP.

In order to apply the PMP to the problem $(\mathbf{P}_L)$, we define the following Hamiltonian function:

$$h(\lambda, p, q, u) = \xi f(q) + \zeta u + \lambda L(u),$$

with $p = (\xi, \zeta) \in T_q^* Q$ and $\lambda \leqslant 0$.

An extremal curve is by definition a quadruple $(\lambda, p(\cdot), q(\cdot), u(\cdot))$, meeting the following necessary conditions for optimality.

We recall that the PMP states that if $q(\cdot)$ is an optimal trajectory corresponding to a control $u(\cdot)$ defined on an interval $[0, T]$, then there exist an absolutely continuous curve $p(\cdot) : [0, T] \to T^*_{q(\cdot)} Q$ and $\lambda \leqslant 0$ such that the pair $(\lambda, p(t))$ never vanishes and for a.a. $t \in [0, T]$, $(q(\cdot), p(\cdot))$ satisfies the Hamiltonian system:

$$\begin{cases} \dot{q}(t) = \dfrac{\partial h}{\partial p}(\lambda, p(t), q(t), u(t)) \\ \dot{p}(t) = -\dfrac{\partial h}{\partial q}(\lambda, p(t), q(t), u(t)), \end{cases} \tag{3.1}$$

and the maximality condition:

$$h(\lambda, p(t), q(t), u(t)) = \max_{v \in \mathbb{R}} h(\lambda, p(t), q(t), v). \tag{3.2}$$

Also, since the final time is free, the Hamiltonian is identically zero along the optimal trajectory, namely,

$$\xi(t) f(q(t)) + \lambda L(u(t)) + \zeta(t) u(t) = 0. \tag{3.3}$$

If $\lambda \neq 0$, the extremal is called *normal*, while if $\lambda = 0$, the extremal is called *abnormal*. Notice that abnormal extremals are extremal (and often optimal) for any cost since in this case the cost disappears from the Hamiltonian.

Remark 3.4. Here we have to face problems, due to the fact that abnormal trajectories come unavoidably in the picture:

- abnormal extremals, if they are not normal at the same time, cannot provide any information on the system. Hence we have to exclude them from our study;
- abnormals that are normal at the same time are automatically smooth (Lem. 3.5 below). Hence we can assume smoothness of our experiments.
- In fact, and for a purely technical reason, certain trajectories that are not abnormal cause problems: those that are abnormal on a piece of themselves only. We shall also exclude them, however, this exclusion can certainly be avoided.

3.1.3. Description of normal extremals of ($\mathbf{P}_L$)

In this case, since $\lambda \neq 0$ we can normalize and assume that $\lambda = -1$.

In this case, the maximality condition (3.2) rewrites

$$h(\lambda, p(t), q(t), u(t)) = \max_{v \in \mathbb{R}} h(-1, p(t), q(t), v) = \xi(t) f(q(t)) + L^*(\zeta(t)),$$

with $L^*(\zeta) = \max_u(\zeta u - L(u))$ being the Legendre transform of L. The properties of the Legendre transform imply the following relations between the optimal control $u(\cdot)$ and $\zeta(\cdot)$:

$$u(t) = {L^*}'(\zeta(t)), \quad \zeta(t) = L'(u(t)), \tag{3.4}$$

so that the maximality condition (3.3) may be rewritten as

$$L(u(t)) - u(t)L'(u(t)) = \xi(t) f(q(t)). \tag{3.5}$$

The equation for q in (3.1) is nothing but (1.3), while the equation for p, also called the *adjoint equation*, can be explicitly rewritten using ξ and ζ. Summing up, all the information is contained in the system below:

$$\begin{cases} \dot{x} = f(x, y) \\ \dot{y} = u = {L^*}'(\zeta) \\ \dot{\xi} = -\xi \dfrac{\partial f}{\partial x}(x, y) \\ \dot{\zeta} = -\xi \dfrac{\partial f}{\partial y}(x, y) \\ L^*(\zeta) + \xi f(x, y) = 0. \end{cases} \tag{3.6}$$

3.1.4. Smoothness of (normal) optimal controls

Lemma 3.5. *The optimal controls $u(t)$ corresponding to normal trajectories of ($\mathbf{P}_L$) are smooth.*

Proof. Assumption **A2** implies that ${L^*}'$ is a smooth function. Hence, u is smooth as a component of the solution of a smooth differential system. ☐

Remark 3.6. Here is a place where the strong strict convexity is important. This assumption also implies that the set of costs under consideration is an open set. Also, it will be crucial in the normalizations of our costs later, and at the end in the possibility to determine uniquely the initial covectors associated with our experiments.

A by-product of Lemma 3.5 is that in fact our extremal controls are also extremals of the standard ($\mathbb{L}^\infty_{\mathrm{loc}}$) PMP (see [1, 15]).

3.1.5. Characterization of abnormal extremals of ($\mathbf{P}_L$)

As we have already said, certain abnormal extremals will come in the picture. Then we shall characterize them.

First, let us introduce the resolvent $R(t, t_0)$ of the time-varying linear system (3.6) (third equation) to be the matrix solution to $\dot{R} = -R\frac{\partial f}{\partial x}$, $R(t_0, t_0) = \mathrm{Id}_{\mathbb{R}^n}$ (when $t_0 = 0$ we shall write $R(t)$ instead of $R(t, 0)$), and define the mapping V through the following lemma.

Lemma 3.7. *To every monotonic smooth trajectory $(q_0, u(\cdot))$, $q_0 = (x_0, y_0)$, we can associate a well-defined smooth map $V(\cdot) : U \to T_{x_0}X$, defined over the range U of $t \mapsto u(t)$, such that for every covector ξ_0 in $\mathbb{R}^n$ and every u that belongs to the value set of the control, we have*

$$\xi_0 V(u) = \xi(t(u)) f(q(t(u))) \tag{3.7}$$

Proof. Since the covector $\xi(t)$ satisfies the linear differential equation $\dot\xi = -\xi\frac{\partial f}{\partial x}$, $\xi(t) = \xi_0 R(t,t_0)$. Obviously, the matrix R depends only on the experiment so that we can write $(\xi f)(t(u)) = \xi_0 V(u)$, with $V(u) = R(t(u),t_0) f(q(t(u)))$. □

Lemma 3.8. *The (monotonic smooth) trajectory $t \mapsto q(t) = (x(t), y(t))$, is the projection of an abnormal extremal if and only if* $\operatorname{span}\{V(u), u \in U\} \neq T_{x_0}X \simeq \mathbb{R}^n$ *(or, equivalently, if and only if* $\operatorname{span}\{V(u(t)), t \in [0,T]\} \neq \mathbb{R}^n$*).*

Proof. Assume first that $q(\cdot)$ is the projection of an abnormal extremal. Therefore, and since $\lambda = 0$, there exists a non trivial covector $p(\cdot) = (\xi(\cdot), \zeta(\cdot))$ such that

$$h(0, p(t), q(t), u(t)) = \xi(t) f(q(t)) + \zeta(t) u(t) = \max_{v \in \mathbb{R}} h(0, p(t), q(t), v).$$

The maximum being reached in the above equation, we necessarily have $\zeta(t) = 0$ for all $t \in [0,T]$, and then, $\xi(t) \neq 0$ for every $t \in [0,T]$. Consequently, the zero-hamiltonian condition reads $\xi(t) f(q(t)) = 0$, for all $t \in [0,T]$ or equivalently, for all $u \in U$,

$$\xi(0) V(u) = 0, \tag{3.8}$$

with $\xi(0) \neq 0$. This last equation (3.8) exactly means that $\{V'(u), u \in U\}$ does not span $T_{x_0}X$.

Conversely, assume that $\{V(u), u \in U\}$ does not span $T_{x_0}X$. Thus, there exists $\xi_0 \in T^*_{x_0}X$ such that $\xi_0 V(u) = 0$ for every $u \in U$, or, equivalently, that for all $t \in [0,T]$,

$$\xi(t) f(q(t)) = 0, \tag{3.9}$$

with $\xi(\cdot)$ being the solution to the Cauchy problem $\dot\xi = -\xi\frac{\partial f}{\partial x}(q(t))$, $\xi(0) = \xi_0$. Differentiating this last equation (3.9) with respect to t leads to

$$\begin{aligned} 0 &= \dot\xi(t) f(q(t)) + \xi(t)\left(\frac{\partial f}{\partial x}(q(t))\dot x + \frac{\partial f}{\partial y}(q(t))\dot y\right) \\ &= \xi(t)\frac{\partial f}{\partial y}(q(t))\dot y \\ &= -\dot\zeta(t) u(t), \end{aligned}$$

which implies that $\dot\zeta(\cdot)$ vanishes identically (by strict monotonicity, the control function $u(\cdot)$ vanishes at most once on $[0,T]$). Hence, the optimal trajectory $t \mapsto q(t)$ is the projection onto $X \times Y$ of the abnormal extremal $t \mapsto (0, \xi(t), 0, q(t), u(t))$. This ends the proof. □

Lemma 3.9. *Let $\gamma : I \to \mathbb{R}^N$ be a smooth curve, defined on some nontrivial interval I. Then, the two following assertions are equivalent.*

1. *For any subinterval $J \subset I$,* $\operatorname{span}\{\gamma(t), t \in J\} = \mathbb{R}^N$*,*
2. *The set of t_0 such that* $\operatorname{span}\{\gamma(t_0), \gamma'(t_0), \dots, \gamma^{(N-1)}(t_0)\} = \mathbb{R}^N$ *is (open) dense in I.*

Proof. The direct implication: we prove the result by contraposition. Assume that on some open subinterval $\tilde I \subset I$, $\operatorname{span}\{\gamma(t), \dots, \gamma^{(N-1)}(t)\} \neq \mathbb{R}^N$. For every integer i, define

$$r(i) = \sup_{t \in \tilde I} \operatorname{rank}\{\gamma(t), \dots, \gamma^{(i)}(t)\},$$

and let k be the first integer for which $r(k) \neq k+1$, namely $r(k) = k$. Let $J \subset \tilde I$ be a subinterval on which $\operatorname{span}\{\gamma(t), \dots, \gamma^{(k-1)}(t)\} = \mathbb{R}^k$. For every $t \in J$, there exist $\alpha_0(t)$, $\alpha_1(t)$, …, $\alpha_{k-1}(t)$ such that

$$\gamma^{(k)}(t) = \sum_{i=0}^{k-1} \alpha_i(t)\gamma^{(i)}(t). \tag{3.10}$$

Notice that the coefficients $\alpha_i(\cdot)$'s are smooth on J. Indeed, relation (3.10) may be rewritten in matrix form as

$$\gamma^{(k)}(t) = \left(\gamma(t), \dots, \gamma^{(k-1)}(t)\right) \begin{pmatrix} \alpha_0(t) \\ \vdots \\ \alpha_{k-1}(t) \end{pmatrix},$$

which can be smoothly solved in the $\alpha_i(t)$'s $(i = 1, \dots, k-1)$ since $\operatorname{rank}\{\gamma(t), \dots, \gamma^{(k-1)}(t) \mid t \in J\} = k$. Let $t_0 \in J$ and let $\tilde{R}(t, t_0)$ be the resolvent of the linear differential equation (3.10), *i.e.*, $\tilde{R}(t, t_0)$ is the matrix solution to the Cauchy problem $\frac{\mathrm{d}}{\mathrm{d}t}\tilde{R}(t, t_0) = A(t)\tilde{R}(t, t_0)$, $\tilde{R}(t_0, t_0) = \mathrm{Id}_{\mathbb{R}^k}$, with

$$A(t) = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & \dots & 0 & 1 \\ \alpha_0 & \dots & \dots & \dots & \alpha_{k-1} \end{pmatrix}.$$

We have

$$\left(\gamma(t), \gamma^{(k-1)}(t)\right) = \left(\gamma(t_0), \gamma'(t_0), \dots, \gamma^{(k-1)}(t_0)\right) \tilde{R}(t, t_0)^*,$$

with $\tilde{R}(t, t_0)^*$ being the transpose of $\tilde{R}(t, t_0)$. Consequently, for all $t \in J$,

$$\gamma(t) = \sum_{i=1}^{k} \tilde{R}_1^i(t, t_0)\gamma^{(i-1)}(t_0),$$

with $\tilde{R}_1^i(t, t_0)$ being the i-th element of the first line of matrix $\tilde{R}(t, t_0)$. Therefore, $\gamma(J) \subset \mathbb{R}^{k-1}$ and consequently $\gamma(\cdot)$ does not span $\mathbb{R}^N$ in restriction to J.

We now prove the converse. If γ does not span $\mathbb{R}^N$ in restriction to some nontrivial interval J, there exists $k < N$ and a k-dimensional subspace $E^k \subset \mathbb{R}^N$ such that $\gamma(J) \subset E^k$. Consequently, for every $t \in J$ and every integer i, we have $\gamma^{(i)}(t) \in E^k$. This completes the proof. □

Definition 3.10 (normal experiment)**.** A monotonic experiment $(q(\cdot), u(\cdot))$, defined on some time interval $[0, T]$ is normal if $q(\cdot)$ is not the projection of an extremal which is abnormal on $[0, T]$.

Definition 3.11 (normal-regular experiment)**.** A monotonic experiment $(q(\cdot), u(\cdot))$, defined on some time interval $[0, T]$ is normal-regular if $q(\cdot)$ is not the projection of an extremal which is abnormal on some (nontrivial) subinterval of $[0, T]$.

Remark 3.12. Note that a normal experiment maybe non normal-regular.

By Lemma 3.8, the experiment is normal-regular if $V(u)$ (resp. $V(t)$) spans $\mathbb{R}^n$ in restriction to any nontrivial subinterval of $[u_0, u_T]$ (resp. $[0, T]$).

As we said, we want to avoid trajectories such that a piece of them is abnormal, (*i.e.*, non normal-regular experiments). An immediate consequence of Lemma 3.9 is the following corollary (one has to apply Lem. 3.9 to the curve $u \mapsto V(u)$).

Corollary 3.13. *The (monotonic) trajectory, $t \mapsto q(t)$ is normal-regular if and only if the set of u such that* $\operatorname{span}\{V(u), \dots, V^{(n-1)}(u)\} = \mathbb{R}^n$ *is (open) dense in* $[u_0, u_T]$.

3.1.6. Test of abnormality

A reasonable practical test of abnormality is the standard Gram-matrix test. It may be done either using $V(t)$ or $V(u)$. We will write $V(\tau)$ to emphasize the fact that both can be indifferently considered. Define the Gram matrix relative to the interval $[\tau_1, \tau_2]$ as:

$$G_a = \int_{\tau_1}^{\tau_2} V(\tau)V^*(\tau)\mathrm{d}\tau,$$

where V^* denotes the transpose of V.

It is clear that $\mathrm{span}\{V(\tau), \tau \in J\} = \mathbb{R}^N$ iff the symmetric matrix G_a is positive definite, iff its smallest eigenvalue is strictly positive. Then, the test resumes to decide whether a matrix is positive definite which is standard in practice (see *e.g.* [16] and Ref. [6] therein).

Differentiating $\xi_0 V(t)$ as was already done above, we get $0 = \xi_0 V'(t) = \xi_0 R(t)\frac{\partial f}{\partial y}(q(t))$. This leads to the standard fact that linearization is not controllable along abnormals.

3.2. Study of the inverse optimal control problem

3.2.1. Well-posedness of the inverse control problem

The next lemma will show that the inverse control problem ($\reflectbox{$\mathbf{P}_L$}$) is an ill-posed problem.

Lemma 3.14. *Assume that $L \in \mathcal{L}$. Let a, b be arbitrary real numbers with $a > 0$. Set $\tilde{L}(u) = aL(u) + bu$. An experiment relative to L is also an experiment relative to $\tilde{L}$.*

Proof. For abnormal experiments the result is obvious.

Let $(q(\cdot), u(\cdot))$ be a (normal) experiment relative to L. Then, there exists a covector $p(\cdot) = (\xi(\cdot), \zeta(\cdot))$ such that the extremal $(-1, p(\cdot), q(\cdot), u(\cdot))$ is solution to system (3.6).

Consequently, since $L^*(\zeta) = \sup_u(\zeta u - L) = \frac{1}{a}\sup_u((a\zeta + b)u - \tilde{L}) = \frac{1}{a}\tilde{L}^*(a\zeta + b)$, one easily infers that the extremal $(-1, \tilde{p}(\cdot), q(\cdot), u(\cdot))$, with $\tilde{p} = (\tilde{\xi}, \tilde{\zeta}) = (a\xi, a\zeta + b)$, satisfies

$$\begin{cases} \dot{x} = f(x,y) \\ \dot{y} = u = \tilde{L}^{*\prime}(\tilde{\zeta}) \\ \dot{\tilde{\xi}} = -\tilde{\xi}\dfrac{\partial f}{\partial x}(x,y) \\ \dot{\tilde{\zeta}} = -\tilde{\xi}\dfrac{\partial f}{\partial y}(x,y) \\ \tilde{L}^*(\tilde{\zeta}) + \tilde{\xi} f(x,y) = 0, \end{cases}$$

showing that $(q(\cdot), u(\cdot))$ is an experiment relative to $\tilde{L}$. □

3.2.2. Normalization of the cost function

Lemma 3.14 shows that the projections onto the base manifold of the extremal trajectories of problems $(\mathbf{P}_L)$ and $(\mathbf{P}_{aL+bu})$ are the same. In particular it says that the cost reconstruction is an ill-posed problem.

One can only expect to reconstruct a class representative of the cost function under the equivalence relation $\sim$, $\tilde{L} \sim L$ if there exists a real $a > 0$ and a real b such that $\tilde{L}(u) = aL(u) + bu$. It is worth mentioning that under this equivalence relation the minimum of the cost function is not preserved but the set of control values such that $L(u) - uL'(u) = 0$ is preserved (indeed, due to assumption **A1**, **A2**, **A3**, there exist exactly two values for which this relation holds: draw the tangents to the graph of L through the origin).

The following lemma gives a way to select a special class representative for the cost function.

Lemma 3.15 (resetting lemma)**.** *At u_0 the cost function L to be reconstructed can be normalized so that: $L'(u_0) = 0$ and $L''(u_0) = 1$.*

Proof. Let L be a class representative of the cost we aim to reconstruct. In order to get the required normalization, we need to solve for a, b real numbers, with $a > 0$ the following system

$$\begin{pmatrix} L'(u_0) & 1 \\ L''(u_0) & 0 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

which is always possible for a cost function L in $\mathcal{L}$: the coefficient a obtained in this way is positive ($a = 1/L''(u_0)$) due to the strong convexity hypothesis. $\square$

Remark 3.16. Notice that our normalization does not preserve the set $\mathcal{L}$: after normalization, the assumption **A1** may not hold, but this will be of no importance in the following. In fact the right way to define the class $\mathcal{L}$ would be to replace the positiveness of L by the weaker $L(0) > 0$. But this is unnatural for the purpose of proving the existence of minimizers (Thm. 3.1).

3.2.3. Existence and uniqueness of the reconstructed cost

Lemma 3.17. *Fix a system f in $\mathcal{F}$ and a smooth admissible trajectory $(q(\cdot), u(\cdot))$, with $u(\cdot)$ monotonic and defined on some time interval $[t_1, t_2]$. Fix a covector $\xi_0 \in \mathbb{R}^n \setminus \{0\}$. Then, there exists a unique smooth function L defined over the range of $t \mapsto u(t)$, and such that $L'(u(t_0)) = 0$ for some arbitrary point $t_0 \in [t_1, t_2]$, for which the pair $(q(\cdot), u(\cdot))$ is the projection of a normal extremal of problem ($\mathbf{P}_L$) associated with the covector $(\xi(\cdot), \zeta(\cdot))$, $(\xi(t_0), \zeta(t_0)) = (\xi_0, 0)$.*

Proof. Choose $t_0 \in [t_1, t_2]$ and set $u_0 = u(t_0)$, $q_0 = q(t_0)$. Let $V(\cdot)$ be the function associated to $(q(\cdot), u(\cdot))$ and defined over the range of $u(\cdot)$ accordingly to Lemma 3.7.

We begin with the existence of L and we also give a formula for it. The range of u is $[u_1, u_2]$. Assume $u_0 \neq 0$. Consider any $a \in [u_1, u_2]$. The reader can easily check that

$$L(u) = \xi_0 V(u_0) + (\xi_0 V(u_0) - \xi_0 V(a)) \left(\frac{u}{u_0} - 1 \right) + u \int_{u_0}^{u} \frac{\xi_0 V(a) - \xi_0 V(v)}{v^2} \mathrm{d}v, \tag{3.11}$$

is such that $L'(u_0) = 0$, and satisfies

$$L(u) - uL'(u) = \xi_0 V(u). \tag{3.12}$$

If $0 \notin [u_1, u_2]$, this $L(u)$ is smooth and is the unique solution of the standard ODE (3.12) with $L(u_0) = \xi_0 V(u_0)$, $L'(u_0) = 0$. If $0 \in [u_1, u_2]$, let us chose $a = 0$. Define the function g by $g(u) = \xi_0 V(0) - \xi_0 V(u)$. We have

$$g'(u) = -ut'(u)\xi(t(u)) \frac{\partial f}{\partial y}(q(t(u))).$$

Hence, $g(0) = g'(0) = 0$, and we may write, $g(u) = u^2 \tilde{g}(u)$, with $\tilde{g}$ smooth. $L(u) = \xi_0 V(u_0) + (\xi_0 V(u_0) - \xi_0 V(0)(\frac{u}{u_0} - 1) + u \int_{u_0}^{u} \tilde{g}(v) \mathrm{d}v$ is a smooth solution of (3.12), with again $L(u_0) = \xi_0 V(u_0)$, $L'(u_0) = 0$.

Suppose now that $u_0 = 0$. Then, set $L(u) = \xi_0 V(u_0) + u \int_0^u \frac{\xi_0 V(0) - \xi_0 V(v)}{v^2} \mathrm{d}v$. Again, $L(u) - uL'(u) = \xi_0 V(u)$, $g(u) = u^2 \tilde{g}(u)$, with $\tilde{g}$ smooth. $L(u) = \xi_0 V(u_0) + u \int_0^u \tilde{g}(v) \mathrm{d}v$, is a smooth function that meets $L(0) = \xi_0 V(u_0)$, $L'(0) = 0$.

Note that the solution $L(u)$ is well defined and smooth over the full interval $[u_1, u_2]$.

We now prove uniqueness, when $0 \in [u_1, u_2]$. We assume that $0 \in [u_0, u_2]$ (the case $0 \in [u_1, u_0]$ is similar).

Let L_1 and L_2 be two solutions to equation (3.12) such that $L_1'(u_0) = L_2'(u_0) = 0$. Then, $\psi = L_1 - L_2$ is a C^∞ solution to

$$\psi(u) - u\psi'(u) = 0, \quad \psi(u_0) = 0, \quad \psi'(u_0) = 0.$$

If u_0 is nonzero, then ψ (a smooth function) is identically zero on $[u_0, 0]$. On $]0, u_2]$, we must have $\psi(u) = Ku$, and since we look for a smooth function, $\psi'(u) = K$ must be zero. At the end, $\psi = 0$, which proves uniqueness.

It remains to prove that the pair $(q(\cdot), u(\cdot))$ is the projection of an extremal trajectory of the problem $(\mathbf{P}_L)$. By hypothesis, the given trajectory is an admissible trajectory of system (1.3).

By definition, $V(u(t)) = R(t, t_0) f(q(t))$. Set $\xi(t) = \xi_0 R(t, t_0)$ so that $\xi(\cdot)$ satisfies the third equation of system (3.6).

Set $\zeta(t) = L'(u(t))$. Thus, $\zeta(t_0) = 0$, and

$$\begin{aligned}
\dot{\zeta}(t) &= L''(u(t))\dot{u}(t) \\
&= -\xi_0 \frac{V'(u)}{u} \dot{u}(t) \\
&= -t'(u)\xi(t(u)) \frac{\partial f}{\partial y}(q(t(u)))\dot{u}(t) \\
&= -\xi(t(u)) \frac{\partial f}{\partial y}(q(t(u))),
\end{aligned}$$

which shows that $\zeta(\cdot)$ satisfies the fourth equation of system (3.6).

By construction equation (3.5) is satisfied. This ends the proof. □

Remark 3.18. Note that it could be that the trajectory under consideration in the lemma is abnormal. In that case, by (3.11), the smooth L reconstructed in the lemma is a constant.

3.3. Validation of the hypotheses

3.3.1. Validation of the hypothesis $L(u)$

We strongly think that the best way to numerically validate the hypothesis $L = L(u)$ is to solve the set of equations (4.3) in the least squares sense. The interest of the least squares approach is that it allows an arbitrary number of experiments (strictly larger than 2). The least squares solution provides adjoint initial vectors, that can be used to reconstruct the cost over the set of values of u visited during all experiments (provided that they overlap).

Then as soon as the experiments are compatible (*i.e.*, the trajectories are actually solutions of an optimal control problem of the required form), the least squares procedure provides a solution. Moreover, if the answer is positive, we have a constructive way to reconstruct the cost.

3.3.2. Testing the strong convexity of the cost

Once the covector ξ_0 has been found, it is easy to check the strong convexity of the cost function. Indeed, differentiating equation (3.5) with respect to u leads to

$$uL''(u) + \xi_0 V'(u) = 0.$$

Thus, when $u \neq 0$, the cost function is strongly convex at u if $\xi_0 V'(u) < 0$. At $u = 0$, it is enough to differentiate equation (3.5) once more to see that the cost function is strongly convex at zero if $\xi_0 V''(0) < 0$.

Consequently, if the system of equations (4.3) admits a solution, it is reasonable to add one of the two above inequalities in order to validate the strong convexity of the cost.

4. The results

4.1. Statement of the main theorem

The inverse control problem being properly posed, we can state our main theoretical result. To state it, let us endow the set $\mathcal{F}$ of systems under consideration with the C^∞ Whitney topology.

Theorem 4.1. There is a residual set of systems in $\mathcal{F}$ such that three monotonic, different, compatible and normal-regular experiments which visit common control values are enough to reconstruct the cost function L over the set of values of u visited within the three experiments. In other terms, cost reconstruction with three experiments (whose range of controls overlap) is a generic property. Moreover, generically, as shown in Theorem 4.15, two experiments may not be enough.

Remark 4.2. Of course, it does not mean that, in practice experiments are monotonic. A non monotonic experiment consists of several monotonic pieces. For instance, an experiment which visits three times the same value of the control is generally sufficient for the cost reconstruction.

The proof of Theorem 4.1 is postponed to Section 4.3, after certain preliminaries given in the next section.

4.2. Preliminaries for the proof of Theorem 4.1

4.2.1. Notations

We will use the following notations in the remaining of the paper:

- $Q_{(\ell)} = \{q_{(\ell)} = (q^1, \dots, q^\ell) \mid q^\alpha \neq q^\beta \quad \text{for} \quad 1 \leqslant \alpha < \beta \leqslant \ell\}$.
- $\mathbb{R}_*^{k\ell+1}$ denotes the set of typical ℓ-tuples $(\mathcal{U}_k^1, \dots, \mathcal{U}_k^\ell)$ of k-jets of controls, $\mathcal{U}_k^i = (u_0^i,\ u_1^i = \dot{u}^i(0),\ \dots,\ u_k^i = u^{i(k)}(0))$, meeting $u_0^i = u_0 \neq 0$, $u_1^1 \neq 0, \dots, u_1^\ell \neq 0$ (common nonzero value u_0 of the control at $t = 0$, and first derivatives of the control nonvanishing at $t = 0$).
- $Z_{k,\ell} = Q_{(\ell)} \times \mathbb{R}_*^{k\ell+1}$. Hence, $\dim Z_{k,\ell} = \ell n + 1 + k\ell$. Independently of the values of k and ℓ, an element of $Z_{k,\ell}$ is denoted by z.
- The bundle $(J^k\mathcal{F})_*^\ell$ is the restriction of the product bundle $(J^k\mathcal{F})^\ell$ to $Q_{(\ell)}$, where $J^k\mathcal{F}$ denotes the bundle over Q of k-jets of elements of $\mathcal{F}$. Typical elements of $(J^k\mathcal{F})_*^\ell$ are tuples

$$j^k f_{(\ell)}(q_{(\ell)}) = (j_{q^1}^k f, \dots, j_{q^\ell}^k f), \quad q_{(\ell)} = (q^1, \dots, q^\ell) \in Q_{(\ell)}.$$

4.2.2. Estimation of the number of needed experiments for the cost reconstruction

For $f \in \mathcal{F}$, let $(q^1(\cdot), u^1(\cdot)), \dots, (q^\ell(\cdot), u^\ell(\cdot))$ be ℓ $(\ell > 1)$ compatible, monotonic, different and normal-regular experiments to which correspond initial covectors $\xi_0^1, \dots, \xi_0^\ell$ respectively. Denote by $U^1, \dots, U^\ell$ the range values of $u^1(\cdot), \dots, u^\ell(\cdot)$ respectively.

Assume that these experiments visit a common control value u_0; and thus, by monotonicity, visit a common control open interval $U \subset \mathbb{R}$. Without loss of generality, we may assume that

- $u_0 \neq 0$;
- $q^\alpha(0) \neq q^\beta(0)$ for $1 \leqslant \alpha < \beta \leqslant \ell$ (since the experiments are different);
- $u_0 = u^1(0) = \dots = u^\ell(0)$ (up to a translation of time on each experiment);
- $\dot{u}^1(0) \neq 0, \dots, \dot{u}^\ell(0) \neq 0$.
- All the matrices $A_k^\alpha = \Big(V^\alpha(u_0), V^{\alpha\prime}(u_0), \dots, V^{\alpha(k)}(u_0)\Big)$ have full rank, where $V^\alpha(\cdot)$ is the map defined in Lemma 3.7.

Remark 4.3. - The reason why we assume that $u_0 \neq 0$ will be made clear later, in the proof of Lemma 4.12. It is related with the fact that, when $u_0 = 0$ (which is possible), in formula (4.10), Lemma 4.12, $\xi_0^\alpha V^{\alpha\prime}(u_0)$ appears multiplied by u_0.

- The fourth assumption is justified by (strong) monotonicity of the experiments.

- The fifth assumption is a consequence of the fact that the experiments are normal-regular and of Corollary 3.13.

- The reason for the second among the five preceding items, will be made clear later, in the proof of Corollary 4.13. It is related to the fact that we will need to fix k-jets of the same function at different points simultaneously.

On the interval U of common control values, the following holds:

$$\xi_0^1 V^1(u) = \cdots = \xi_0^\ell V^\ell(u) = L(u) - uL'(u), \tag{4.1}$$

and the non homogeneous equation

$$\xi_0^1 {V^1}'(u_0) = \cdots = \xi_0^\ell {V^\ell}'(u_0) = -u_0 \ (\neq 0), \tag{4.2}$$

has to be considered if we wish to reconstruct the cost function normalized according to Lemma 3.15.

Equations (4.1) and (4.2) imply that for all integers $k \geqslant 0$

$$\begin{cases} (\xi_0^1, \ldots, \xi_0^\ell) M(z) = 0, \\ \xi_0^1 {V^1}'(u_0) = -u_0, \end{cases} \tag{4.3}$$

where $M(z)$ is the $\ell n \times (\ell-1)(k+1)$ matrix defined by

$$M(z) = \begin{pmatrix} A_k^1 & 0 & \ldots & 0 \\ -A_k^2 & A_k^2 & \ddots & \vdots \\ 0 & -A_k^3 & \ddots & 0 \\ \vdots & \ddots & \ddots & A_k^{\ell-1} \\ 0 & \ldots & 0 & -A_k^\ell \end{pmatrix}, \tag{4.4}$$

with $A_k^\alpha = \Big(V^\alpha(u_0), {V^\alpha}'(u_0), \ldots, V^{\alpha(k)}(u_0)\Big)$ being the k-jet of $V^\alpha(\cdot)$ at u_0, and z has been defined in Section 4.2.1 above.

Remark 4.4. Notice that the equations $\xi_0^\alpha {V^\alpha}'(u_0) = -u_0$ for $\alpha = 2, \ldots, \ell$ are omitted since they follow from the second equation of system (4.3).

The main theorem (Thm. 4.1) is a consequence of the fact that system (4.3) has (generically) a unique solution, as we shall prove.

Before, let us examine the different cases showing up for the solutions of system (4.3). Let $\mathcal{S}_{k,\ell}$ denote the space of $\ell n \times (\ell-1)(k+1)$ matrices of the form (4.4). Assume that $\ell n \leqslant (\ell-1)(k+1)$. This is not a restriction since, later, k will be taken large. The set of matrices $\mathcal{S}_{k,\ell}$ is stratified by the rank, each strata being a submanifold of the set of $\ell n \times (\ell-1)(k+1)$ matrices. Namely,

$$\mathcal{S}_{k,\ell} = \bigcup_{r=0}^{\ell n} \mathcal{S}_{k,\ell}(r),$$

with $\mathcal{S}_{k,\ell}(r)$ being the subset of matrices of corank r.

The three following situations may appear:

$\boldsymbol{M(z) \in \mathcal{S}_{k,\ell}(0)}$. In this case $M(z)$ has full rank. The unique solution to the first equation of system (4.3) is zero and consequently the system (4.4) has no solution. This may happen if: either the considered experiments are not compatible (they are not extremal for the same cost), or the assumptions made on the cost function are not satisfied.

Remark 4.5. This case cannot happen under the assumption that the experiments are monotonic, different and compatible.

$M(z) \in \mathcal{S}_{k,\ell}(1)$. This case is the one we are interested with. We will show immediately that system (4.3) admits a unique solution and then, according to Lemma 3.17 the cost is uniquely determined.
Let us start with $\hat{\xi}_0 = (\xi_0^1, \dots, \xi_0^\ell) \neq 0$ vanishing on $M(z)$. If $\xi_0^\alpha V^{\alpha\prime}(u_0) \neq 0$, the inhomogeneous condition in (4.3) can be satisfied just multiplying ξ_0 by an appropriate nonzero constant, and the solution so obtained to equations (4.3) is unique.
If $\xi_0^\alpha V^{\alpha\prime}(u_0) = 0$, let us show that we can change u_0 (an arbitrarily small change for the 5 open conditions of the beginning of the section still hold true) for $\xi_0^\alpha V^{\alpha\prime}(u_0) \neq 0$.
The experiment being assumed normal-regular, we claim that $\xi_0^\alpha V^{\alpha\prime}(u(t)) \neq 0$ for some t in an open interval arbitrarily close to $t(u_0) = 0$. Indeed, assume that $\varphi^\alpha(t) = \xi_0^\alpha R^\alpha(t) f(q^\alpha(t))$ is constant on some interval (the resolvent $R^\alpha(t)$ has been defined at the beginning of Sect. 3.1.5).
Then, if this constant is zero, it means exactly that the trajectory $q^\alpha(\cdot)$ is abnormal on this neighborhood, which contradicts the normal-regular assumption. If this constant (say k) is non zero, it means that the trajectory is normal and

$$\xi_0^\alpha V^\alpha(u) = L(u) - uL'(u) = k \neq 0,$$

with, in addition, $L'(u_0) = 0$ and $u_0 \neq 0$. It yields (integrating the differential equation) to $L(u) = k$ contradicting the strict convexity of L.
Then, we can take for u_0 a $u^\alpha(t_0)$ such that $\dot{\varphi}^\alpha(t_0) \neq 0$. The new ξ_0^α will be automatically (a scalar multiple of) $\xi_0^\alpha R^\alpha(t_0)$. Moreover, $M(z(t))$ will still be in $\mathcal{S}_{k,\ell}(1)$: it cannot belong to $\mathcal{S}_{k,\ell}(0)$ since the experiments are compatible, and $\mathcal{S}_{k,\ell}$ is stratified by the rank.

$M(z) \in \bigcup_{r \geqslant 2} \mathcal{S}_{k,\ell}(r)$. This case is the one we want to avoid. Indeed, if $\operatorname{corank} M(z) \geqslant 2$, then system (4.3) admits a solution but this solution is not unique and, according to Theorem 4.15, the cost L is not uniquely determined.

In the following we will study this last case and prove (using Thom's transversality theory) that it is generically impossible as soon as $\ell \geqslant 3$ and k is chosen large enough.

4.2.3. The bad sets

Let $\mathcal{M}_{k,\ell}$ be the set of structured matrices of the form (4.4).

Definition 4.6. Let $B_{k,\ell}(r)$ be the subset of $\mathcal{M}_{k,\ell}$ which elements M satisfy:

corank $M = r \geqslant 2$;
every submatrix A_k^α has full rank ($\alpha = 1, \dots, \ell$).

Set $B_{k,\ell} = \bigcup_{r=2}^{\ell n} B_{k,\ell}(r)$.

Lemma 4.7. *If $M \in B_{k,\ell}$, its cokernel has no non zero element of the form $(0, \xi^2, \dots, \xi^\ell)$ or $(\xi^1, 0, \xi^3, \dots, \xi^\ell)$ or … or $(\xi^1, \dots, \xi^{\ell-1}, 0)$.*

Proof. We prove the result by contradiction. Assume without loss of generality that $(0, \xi^2, \dots, \xi^\ell) \in \operatorname{coker} M$. This means that $(0, \xi^2, \dots, \xi^\ell)M = 0$. Consequently, $\xi^2 A_k^2 = 0$, which implies that $\xi^2 = 0$ since A_k^2 as full rank. By induction, we get $\xi^\alpha = 0$ for all $\alpha = 1, \dots, \ell$. □

Let $Gr(r, \mathbb{R}^{\ell n})$ denote the Grassmannian of $\mathbb{R}^{\ell n}$, that is the space of all r-dimensional vector subspaces of $\mathbb{R}^{\ell n}$. With a little abuse a notation, we identify a r-dimensional linear space $\Pi \in Gr(r, \mathbb{R}^{\ell n})$ with any r-tuple vectors $(\hat{\xi}_1, \dots, \hat{\xi}_r) = (\xi_1^1, \dots, \xi_1^\ell, \dots, \xi_r^1, \dots, \xi_r^\ell)$ such that $\Pi = \operatorname{span}(\hat{\xi}_1, \dots, \hat{\xi}_r)$.

Definition 4.8. Let $\widetilde{B}_{k,\ell}(r)$ be the subset of $Gr(r, \mathbb{R}^{\ell n}) \times \mathcal{M}_{k,\ell}$ of all tuples $(\hat{\xi}_1, \dots, \hat{\xi}_r, M)$ such that $M \in B_{k,\ell}$ and $(\hat{\xi}_1, \dots, \hat{\xi}_r)M = 0$.

Notice that the set $B_{k,\ell}$ is the projection of $\widetilde{B}_{k,\ell} = \bigcup_{r=0}^{\ell n} \widetilde{B}_{k,\ell}(r)$ parallel to the Grassmannian $Gr(r, \mathbb{R}^{\ell n})$.

4.2.4. Estimation of the codimension of $B_{k,\ell}$

The Thom's coranks formula (see [17]) does not hold in general for the structured matrices M (*i.e.*, of the form (4.4)), but it still holds for the matrices M whose blocks have full rank. More precisely we have the following lemma.

Lemma 4.9. *The semi-algebraic set $B_{k,\ell}$ satisfies*

$$\operatorname{codim} B_{k,\ell}(r) \geqslant r\left((\ell-1)(k+1)-\ell(n-r)\right), \tag{4.5}$$

where we have assumed that $\ell(n-r)<(\ell-1)(k+1)$.

Proof. Let $M \in B_{k,\ell}(r)$. Then, we can find r independent vectors $\hat{\xi}_1=(\xi_1^1,\ldots,\xi_1^\ell)$, $\ldots$, $\hat{\xi}_r=(\xi_r^1,\ldots,\xi_r^\ell)$ in the cokernel of M, *i.e.*,

$$\xi_i^\alpha A^\alpha - \xi_i^{\alpha+1}A^{\alpha+1}=0, \quad i=1,\ldots,r \quad \alpha=1,\ldots,\ell-1,$$

where, for simplicity, we have set $A^\alpha = A_k^\alpha$ ($\alpha=1,\ldots,\ell$). We may assume without loss of generality that, for $\alpha=1,\ldots,\ell-1$, the ξ_i^α's ($i=1,\ldots,r$) are the rth first vectors of the canonical dual basis of $\mathbb{R}^n$, namely, that $\xi_i^\alpha=(0,\ldots,0,1,0,\ldots,0)$. Indeed, if $\xi_1^1,\ldots,\xi_r^1$ were dependent, there would exist $(\lambda_1,\ldots,\lambda_r)\neq(0,\ldots,0)$ so that $\lambda_1\xi_1^1+\cdots+\lambda_r\xi_r^1=0$. Consequently, we would have a non zero element of the form $(0,\xi_2,\ldots,\xi_r)$ in the cokernel of M contradicting that M belongs to $B_{k,\ell}(r)$ (and the same holds for $\hat{\xi}^\alpha$, $\alpha=2,\ldots,\ell-1$).

Now, in a neighborhood of $(\hat{\xi}_1,\ldots,\hat{\xi}_r,M)$, let us consider the set of $r(\ell-1)(k+1)$ equations

$$F_i^\alpha = \left(\xi_i^\alpha+\delta\xi_i^\alpha\right)\left(A^\alpha+\delta A^\alpha\right)-\left(\xi_i^{\alpha+1}+\delta\xi_i^{\alpha+1}\right)\left(A^{\alpha+1}+\delta A^{\alpha+1}\right)=0, \tag{4.6}$$

(with $i=1,\ldots,r$ and $\alpha=1,\ldots,\ell-1$), which we want to solve in $w=(w_1,w_2,\ldots,w_{r(\ell-1)(k+1)})=(\delta A_1^\alpha,\ldots,\delta A_r^\alpha)$ where δA_i^α denotes the i-th line of δA^α.

Notice that,

$$\frac{\partial F_i^\alpha}{\partial w}(0)=\left.\frac{\partial}{\partial w}\right|_{w=0}F^\alpha(w,0)=\left.\frac{\partial}{\partial w}\right|_{w=0}\delta\xi_i^\alpha\delta A^\alpha-\delta\xi_i^{\alpha+1}\delta A^{\alpha+1}=\left.\frac{\partial}{\partial w}\right|_{w=0}\delta A_i^\alpha-\delta A_i^{\alpha+1}$$

implies that the Jacobian matrix at zero with respect to the variables w of the mapping $F=F_1^1\times F_2^1\times\cdots\times F_r^1\times F_1^2\times\cdots\times F_r^\ell$ associated to the system (4.6) has the form

$$\frac{\partial F}{\partial w}(0)=\begin{pmatrix} \mathrm{Id}_{\mathbb{R}^{k+1}} & * & \ldots & * \\ 0 & \mathrm{Id}_{\mathbb{R}^{k+1}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & * \\ 0 & \ldots & 0 & \mathrm{Id}_{\mathbb{R}^{k+1}} \end{pmatrix}.$$

Hence, by the implicit function theorem, we can smoothly solve system (4.6) in a neighborhood of zero.

Therefore the set of solutions to system (4.6) is a manifold of codimension $r(\ell-1)(k+1)$, namely,

$$\operatorname{codim} \widetilde{B}_{k,\ell}(r)=r(\ell-1)(k+1).$$

Since $B_{k,\ell}(r)$ is the projection of $\widetilde{B}_{k,\ell}(r)$ on $\mathcal{M}_{k,\ell}$ parallel to the Grassmannian $Gr(r,\mathbb{R}^{\ell n})$, we have:

$$\begin{aligned}\operatorname{codim} B_{k,\ell}(r) &\geqslant \operatorname{codim} \widetilde{B}_{k,\ell}(r)-\operatorname{codim} Gr(r,\mathbb{R}^{\ell n})\\ &\geqslant r(\ell-1)(k+1)-r(\ell n-r).\end{aligned}$$

This is the Thom's "product of coranks" bound.

Since $r\geqslant 2$, we get (k large):

$$\operatorname{codim} B_{k,\ell}\geqslant 2((\ell-1)(k+1)-\ell n+2). \qquad \square$$

The following proposition, which is sufficient to prove our main theorem (Thm. 4.1), is a consequence of the considerations of this section.

Proposition 4.10. If $\ell = 3$ and k is large enough, then the set of $f \in \mathcal{F}$ such that the map $z \mapsto M(z)$ avoids $B_{k,3}$, in restriction to $Z_{k,3}$, is residual in the Whitney topology.

Remark 4.11. The estimate $\operatorname{codim} B_{k,\ell} \simeq 2(\ell-1)k$ for k large shows that we cannot expect that two experiments are enough: see Sections 4.3.3 and 4.3.5.

4.3. Technical mathematical tools and final proof

In this last section, besides a technical lemma and the final proof of our main theorem, we show that in general two experiments are not enough.

4.3.1. A crucial lemma and its corollary

Here, $\mathbb{R}_*^{k+1}$ denotes the set of $(k+1)$-tuples $(u_0, u_1, \ldots, u_k)$ with u_0, u_1 both nonzero (k-jets of smooth controls that are nonzero and monotonic). Define the following mapping:

$$\begin{aligned} \Xi : J^k\mathcal{F} \times \mathbb{R}_*^{k+1} &\rightarrow J^k(\mathbb{R}, TX) \\ (j^k f, j_0^k u) &\mapsto j_{u_0}^k V \end{aligned}$$

Lemma 4.12. *The mapping Ξ is a surjective submersion.*

Proof. **Note.** Along this proof, we will make a constant abuse, for simplicity in the notations: we are treating k-jets of systems f and controls $u(\cdot)$, as the system f or the control $u(\cdot)$ themselves. This is justified by the fact that all the manipulations we do depend only on the k-jets of the objects. For instance, the k-jets of the function $t(u)$ depend (smoothly and diffeomorphically) on the k-jets of $u(t)$ only, due to the strict monotonicity, reflected in the definition of $\mathbb{R}_*^{k+1}$.

A point $(u_0, u_1 \ldots, u_k) \in \mathbb{R}_*^{k+1}$ is given, together with a k-jet $j^k f \in J^k\mathcal{F}$ with source $q_0 = (x_0, y_0)$. Then, $(V(u_0), V'(u_0), \ldots, V^{(k)}(u_0))$ is well defined.

Assume that $q = q(t)$ (equivalently $q(u)$) is a (monotonic smooth) trajectory defined on a neighborhood of zero (equivalently u_0) such that $q(0) = q_0$ (equivalently $q(u_0) = q_0$). Let $R(u) = R(t(u))$ be the resolvent defined at the beginning of Section 3.1.5.

Below we shall compute the successive derivatives with respect to u of $V(u)$. Namely,

$$\begin{aligned} V'(u) &= R(u)\pi_{X*}\left(ut'(u)\mathrm{ad}_{F_1}F_0(q)\right), \\ V^{(r)}(u) &= R(u)\pi_{X*}\Big(\left(ut'(u)\right)^r \mathrm{ad}_{F_1}^r F_0(q) - u\left(t'(u)\right)^r \mathrm{ad}_{F_0}^r F_1(q) \\ &\quad -(r-1)\left(t'(u)\right)^{r-1} \mathrm{ad}_{F_0}^{r-1} F_1(q) + u\psi_r(u) + \psi_{r-1}(u)\Big), \qquad r \geqslant 2, \end{aligned} \tag{4.7}$$

with π_{X*} being the tangent map to the canonical projection $\pi_X : X \times Y \to X$ parallel to Y, and where the notation $\psi_r(u)$ means a term linear in Lie brackets (evaluated at q) of length not greater than $r+1$ containing F_0 and F_1 and in which F_0 and F_1 appear at least once ($r \geqslant 1$) and at most $r-1$ times ($r \geqslant 2$). The coefficients of the linear expression $\psi_r(u)$ are functions that belong to $\mathbb{R}[u, t'(u), \ldots, t^{(r)}(u)]$.

Let us now prove formula (4.7). Introduce the following notation for Lie brackets

$$F_I = [F_{i_1}, [F_{i_2}, \ldots [F_{i_{\ell-1}}, F_{i_\ell}] \ldots]], \qquad I = (i_1, \ldots, i_\ell), \qquad |I| = i_1 + \cdots + i_\ell.$$

Define the set $\mathcal{I}_r$ by

$$\mathcal{I}_r = \bigcup_{\ell=1}^{r+1} \left\{ I \in \{0,1\}^\ell \;\middle|\; 1 \leqslant |I| \leqslant r-1,\; 1 \leqslant \ell - |I| \leqslant r-1 \right\}.$$

Then,

$$\psi_r(u) = \sum_{I\in\mathcal{I}_r} c_{I,r}(u)F_I(q) \in TX \times \{0_{TY}\}, \qquad c_{I,r}(u) \in \mathbb{R}[u, t'(u), \dots, t^{(r)}(u)].$$

For any $\xi_0 \in T^*_{x_0}X$, let $p(\cdot) = (\xi(\cdot), \zeta(\cdot))$ be the solution to $\dot{p} = -p\frac{\partial F_0}{\partial q} - up\frac{\partial F_1}{\partial q}$, $p(0) = (\xi_0, 0)$.

Consequently, taking into account that $c_{I,r}(u)'$ is of the form $c_{I,r+1}(u)$ (which is obvious), and that $\psi_r(u) \in TX \times \{0_{TY}\}$, we get

$$\begin{aligned} \xi\pi_{X*}\psi_r'(u) &= p\,\psi_r'(u) && (4.8)\\ &= \sum_{I\in\mathcal{I}_r} c'_{I,r}(u)p\,F_I(q) + c_{I,r}(u)t'(u)\frac{\mathrm{d}}{\mathrm{d}t}\left(p\,F_I(q)\right)\\ &= \sum_{I\in\mathcal{I}_r} c'_{I,r}(u)p\,F_I(q) + c_{I,r}(u)t'(u)p\,\mathrm{ad}_{F_0+uF_1}F_I(q)\\ &= \xi\pi_{X*}\psi_{r+1}(u). && (4.9)\end{aligned}$$

We now prove formula (4.7) recursively. We have

$$\begin{aligned} \xi_0 V'(u) &= \left(\xi\pi_{X*}F_0(q)\right)'(u)\\ &= \left(p\,F_0(q)\right)'(u)\\ &= \frac{\mathrm{d}}{\mathrm{d}t}\left(p\,F_0(q)\right)t'(u)\\ &= p\,\mathrm{ad}_{F_0+uF_1}F_0(q)t'(u)\\ &= \xi\pi_{X*}\mathrm{ad}_{F_1}F_0(q)ut'(u), && (4.10)\end{aligned}$$

which gives the formula for $r = 1$.

Differentiation of formula (4.10) with respect to u gives the formula for $r = 2$:

$$\begin{aligned} \xi_0 V''(u) &= \left(p\,\mathrm{ad}_{F_1}F_0(q)ut'(u)\right)'(u)\\ &= \frac{\mathrm{d}}{\mathrm{d}t}\left(p\,\mathrm{ad}_{F_1}F_0(q)\right)t'(u)ut'(u) + p\,\mathrm{ad}_{F_1}F_0(q)\left(ut'(u)\right)'\\ &= \xi\pi_{X*}\Big(\mathrm{ad}_{F_0+uF_1}\mathrm{ad}_{F_1}F_0(q)t'(u)ut'(u) + \mathrm{ad}_{F_1}F_0(q)\left(ut'(u)\right)'\Big)\\ &= \xi\pi_{X*}\Big(\left(ut'(u)\right)^2\mathrm{ad}^2_{F_1}F_0(q) - u\left(t'(u)\right)^2\mathrm{ad}^2_{F_0}F_1(q) - t'(u)\mathrm{ad}_{F_0}F_1(q) + u\psi_2(u) + \psi_1(u)\Big),\end{aligned}$$

with $\psi_2(u) = t''(u)\mathrm{ad}_{F_1}F_0(q)$, and $\psi_1(u) = 0$.

Now, differentiation of formula (4.7) with respect to u and consideration of equation (4.9), leads straightforwardly to formula (4.7) at rank $r + 1$.

Note that since $F_0 = (f, 0)$, $F_1 = (0, 1)$, we can rewrite the formula (4.7) at $u = u_0$ as:

$$\frac{\mathrm{d}^r V}{\mathrm{d}u^r}(u_0) = (u_0 t'(u_0))^r \frac{\partial^r f}{\partial y^r}(q_0) + \phi_r(u_0),$$

where $\phi_r(u_0)$ depends on successive derivatives of f with respect to y up to order strictly lower than r. And since u_0, $t'(u_0)$ are both nonzero, the result follows. □

Corollary 4.13. *The mapping*

$$\begin{aligned} \Xi^\ell : (J^k\mathcal{F})^\ell_* \times \mathbb{R}^{k\ell+1}_* &\to \mathcal{M}_{k,\ell}\\ (j^k f_{(\ell)}, \mathcal{U}^1_k, \dots, \mathcal{U}^\ell_k) &\mapsto M(z),\end{aligned}$$

with $z = (q^1, \dots, q^\ell, \mathcal{U}^1_k, \dots, \mathcal{U}^\ell_k)$, *is a surjective submersion.*

Proof. The proof is an immediate consequence of Lemma 4.12, since for every $q_{(\ell)} \in Q_{(\ell)}$, the ℓ points $q^1, \dots, q^\ell$ are distinct. □

4.3.2. Proof of Theorem 4.1

In this section, we will apply a slight modification of the standard multijet transversality theorem (see *e.g.* [12, Thm. 4.13]). This theorem is stated for bundles of jets of smooth mappings, but it holds also for jet bundles of sections of a bundle. This is known, and completely clear, since the arguments are essentially local, and locally, a section of a bundle is just a smooth mapping to the fiber.

We state the theorem in our own context only:

Theorem 4.14. Let W be a stratified subset of $(J^k\mathcal{F})^\ell_*$. Let $T_W = \{f \in \mathcal{F} \mid j^k f_{(\ell)} \pitchfork W\}$. Then, T_W is a residual subset of $\mathcal{F}$ for the Whitney topology.

4.3.3. Definition of $N_{k,\ell}$

The set $N_{k,\ell} \subset J^k\mathcal{F}_{(\ell)}$ that we will construct below, will, in some sense, represent the set $B_{k,\ell}$ in $J^k\mathcal{F}_{(\ell)}$.

By Corollary 4.13, the map Ξ^ℓ is transversal to the points. Therefore, $(\Xi^\ell)^{-1}(B_{k,\ell})$ is a semi-algebraic Whitney-b stratified set of the same codimension $c_{k,\ell}$, *i.e.*, $c_{k,\ell} \geqslant 2((\ell-1)(k+1)-\ell n+2)$ by Lemma 4.9.

Let $\pi_1 : (J^k\mathcal{F})^\ell_* \times \mathbb{R}^{k\ell+1}_* \to (J^k\mathcal{F})^\ell_*$ denotes the canonical projection on the first factor parallel to $\mathbb{R}^{k\ell+1}_*$.

Define the set $N_{k,\ell}$ by:

$$N_{k,\ell} = \pi_1\left((\Xi^\ell)^{-1}(B_{k,\ell})\right).$$

Consequently, $N_{k,\ell}$ is a stratified set of codimension:

$$\begin{aligned}\operatorname{codim} N_{k,\ell} &\geqslant \operatorname{codim} B_{k,\ell} - (k\ell+1)\\ &\geqslant 2((\ell-1)(k+1)-\ell n+2) - k\ell - 1\end{aligned}$$

For $\ell = 3$,

$$\operatorname{codim} N_{k,3} \geqslant k - 6n + 7.$$

4.3.4. End of the proof of Theorem 4.1

By Theorem 4.14, the set $\mathcal{G}$ of $f \in \mathcal{F}$ such that $j^k f_{(3)}$ is transversal to $N_{k,3}$ is residual for the Whitney topology. If k is large enough, it means that $j^k f_{(3)}$ avoids $N_{k,3}$.

But avoiding $N_{k,3}$ means exactly that the map $z \mapsto M(z)$ avoids $B_{k,3}$, in restriction to $Z_{k,3}$. By Proposition 4.10, this ends the proof of Theorem 4.1

4.3.5. Necessity of three experiments

Theorem 4.15. Generically, two experiments are not enough in order to reconstruct the cost.

Proof. Let $(q^A(\cdot), u^A(\cdot))$ and $(q^B(\cdot), u^B(\cdot))$ be two compatible, monotonic, different and normal-regular experiments to which correspond V^A and V^B. According to Lemma 3.9, there exists $u_0 \neq 0$ such that $j^k_{u_0}V^A$ and $j^k_{u_0}V^B$ have full rank. By compatibility of the experiments, there exist ξ^A and ξ^B such that

$$\xi^A j^k_{u_0}V^A - \xi^B j^k_{u_0}V^B = 0, \tag{4.11}$$

$$\xi^A V^{A\prime}(u_0) = -u_0. \tag{4.12}$$

Set $M(z) = (j^k_{u_0}V^A,\ -j^k_{u_0}V^B)^*$ and suppose that $\dim\operatorname{coker} M(z) = 2$ (the strata of matrices $M(z)$ whose cokernels have dimension strictly larger than two is generically avoided, by formula (4.5), but the stratum corresponding to corank 2 is not avoided in general). Then, the set $E = \{(\xi^A, \xi^B) \mid (4.11) \text{ and } (4.12) \text{ are satisfied}\}$ has dimension one. Hence, there exist two independant pairs of covectors (ξ^A_1, ξ^B_1) and (ξ^A_2, ξ^B_2) such that $E = (\xi^A_1, \xi^B_1) + \mathbb{R}(\xi^A_2, \xi^B_2)$.

By (4.12), $\xi^{A,B}_{1,2}$ are all nonzero. Assume that ξ^A_1, ξ^A_2 are linearly dependant, then, some $\hat{\xi}_A = \xi^A_1 + \lambda\xi^A_2 = 0$. Set $\hat{\xi}_B = \xi^B_1 + \lambda\xi^B_2$, and $\hat{\xi} = (0, \hat{\xi}_B) \in E$ for $\hat{\xi}_B \neq 0$. This again is impossible by normal-regularity, since it implies $\hat{\xi}^B j^k_{u_0}V^B = 0$. Therefore, ξ^A_1, ξ^A_2 are linearly independant.

The cost function defined by

$$L(\lambda, u) = (\xi_1^A + \lambda \xi_2^A)\left(V^A(u_0) + u\int_{u_0}^{u} \frac{V^A(u_0) - V^A(v)}{v^2}\mathrm{d}v\right),$$

is clearly a solution of the inverse problem for any λ in $\mathbb{R}$. Thus, $L(\lambda, u)$ depends on λ unless $\frac{\partial L}{\partial \lambda} \equiv 0$.

But, $L(\lambda, u) - u\frac{\partial L(\lambda,u)}{\partial u} = (\xi_1^A + \lambda\xi_2^A)V^A(u)$, hence if $\frac{\partial L}{\partial \lambda} \equiv 0$ then differentiating w.r.t. λ, $0 = \frac{\partial}{\partial \lambda}(L(\lambda, u) - u\frac{\partial L(\lambda,u)}{\partial u}) = \xi_2^A V^A(u)$, which again cannot be identically zero by normal regularity.

Consequently, two experiments do not allow to reconstruct the cost function uniquely. □

5. The inverse optimal control problem for the Dubins system

5.1. Preliminaries

This part is concerned with the resolution of problem $(\reflectbox{$\mathbf{P}_L$})$ for Unmanned Aerial Vehicles (UAVs).

In this study, we consider a HALE (High Altitude Long Endurance) UAV flying at constant speed and altitude. The equations of motion are the ones of the Dubins car (see [9] for a justification), *i.e.*,

$$\begin{cases} \dot{x} = \cos\theta \\ \dot{y} = \sin\theta \\ \dot{\theta} = u, \end{cases} \tag{5.1}$$

with $q = (x, y, \theta) \in \mathbb{R}^2 \times S^1$ being the state (where $(x, y) \in \mathbb{R}^2$ is the UAV's coordinate in the constant altitude plane, and θ the yaw angle), and $u \in \mathbb{R}$ being the control variable.

Inspired from the works [6,8,10] we assume that the trajectories flown by experimented pilots are solutions of some optimal control problem. The problem is to decide what is minimized. As written in [8], this assumption is based on a "nowadays widely accepted paradigm in neurophysiology which says that, among all possible movements, the accomplished ones satisfy suitable optimality criteria (see [14] for a review)". In [8, 10] the authors address the problem of reconstruction of the cost minimized in human locomotion. It's worth to notice that our problem is very similar since HALE drones behave kinematically more or less as a human being moving on a plane (constant altitude, constant speed).

Remark 5.1. 1. Up to a suitable rototranslation and a time shift, we may assume that optimal trajectories meet $q(0) = (0, 0, 0)$ (invariance under motions).

2. Problem $(\mathbf{P}_L)$ for the Dubin's model does not admit any abnormal extremal as the reader can easily check.

3. Note that u is the curvature of the trajectory in the (x, y) plane. A special class of trajectories consists of those trajectories with a single inflexion point (*i.e.*, a zero curvature point). Although our method allows to care about all trajectories, these are of particular interest, as will be discussed in Section 5.3. In that case, it would be completely natural to consider that $u_0 = 0$, as in the papers [8, 10]. In fact, $u_0 = 0$ is a case that we tried to avoid at several places in this paper (we overcome this case by small perturbation). Therefore, we will not make this normalization here.

5.2. Main results in Dubins case

We seek for a cost function normalized as in Lemma 3.15. For system (3.1), the free time condition writes:

$$L(u(t)) - up_\theta(t) = p_x(t)\cos\theta(t) + p_y(t)\sin\theta(t), \tag{5.2}$$

and the adjoint equations are:

$$\begin{cases} \dot{p}_x = 0 \\ \dot{p}_y = 0 \\ \dot{p}_\theta = p_x \sin\theta - p_y\cos\theta. \end{cases} \tag{5.3}$$

Therefore in the remaining of this section, p_x and p_y are real constants.

Taking into account equation (3.4) and the normalization of Lemma 3.15, the evaluation at $t=0$ of the last equation of system (5.3) gives

$$\dot{u}(0) = L''(u(0))\dot{u}(0) = \dot{p}_\theta(0) = -p_y. \tag{5.4}$$

And the last equation of system (5.3) may be rewritten

$$\dot{p}_\theta = p_x \dot{y} - p_y \dot{x}. \tag{5.5}$$

Since $x(0) = y(0) = p_\theta(0) = 0$, we get

$$p_\theta(t) = p_x y(t) - p_y x(t),$$

which, according to (3.4) and (5.2), leads to

$$L(u(t)) = p_x \left(\cos\theta(t) + u(t)y(t)\right) - \dot{u}(0)\left(\sin\theta(t) - u(t)x(t)\right). \tag{5.6}$$

In particular, the cost L is known as soon as the constant p_x is known.

5.2.1. Consider a single experiment $\theta(u)$

Let us consider a single monotonic experiment $\theta(u) = \theta(t(u))$ parametrized by u (the curvature). The choice of any arbitrary p_x determines $L(u)$ according to equation (5.6). Since the $L(u)$ so obtained depends on p_x, we conclude that one experiment is not enough.

5.2.2. Consider two compatible experiments $\theta(u)$, $\tilde{\theta}(u)$

Assume that the respective domains U and $\tilde{U}$ of these experiments have a non void intersection. It is always the case for (x, y) trajectories with an inflexion, to which we can apply the previous normalization $q(0) = (0, 0, 0)$, in particular $\theta(u_0) = \tilde{\theta}(u_0) = 0$, which is responsible for (5.4).

Since the two experiments are compatible we have for all u in $U \bigcap \tilde{U}$

$$\begin{aligned} L(u) - uL'(u) = {} & p_x \cos\theta(u) + p_y \sin\theta(u) \\ = {} & \tilde{p}_x \cos\tilde{\theta}(u) + \tilde{p}_y \sin\tilde{\theta}(u). \end{aligned}$$

Therefore, we must have for all u in U

$$p_x \cos\theta(u) - \tilde{p}_x \cos\tilde{\theta}(u) = -p_y \sin\theta(u) + \tilde{p}_y \sin\tilde{\theta}(u). \tag{5.7}$$

We have the following lemma.

Lemma 5.2. *For two (strictly) monotonic compatible experiments, the set of equations (5.7) determines p_x and $\tilde{p}_x$ as soon as the functions $\cos(\theta(\cdot))$ and $\pm\cos(\tilde{\theta}(\cdot))$ do not coincide on their common domain.*

Proof. For two arbitrary control values u_1, u_2 in U, define the determinant:

$$D(u_1, u_2) = \det\left(\begin{pmatrix} \cos\theta(u_1) \\ \cos\theta(u_2) \end{pmatrix}, \begin{pmatrix} -\cos\tilde{\theta}(u_1) \\ -\cos\tilde{\theta}(u_2) \end{pmatrix} \right).$$

Let us assume that $D(u_1, u_2) = 0$ for all u_1, u_2 in $U \bigcap \tilde{U}$. We deduce that, since θ and $\tilde{\theta}$ play the same role,

$$\exists\ a \in \mathbb{R}, \quad \forall u\ \in U \bigcap \tilde{U} \quad \cos\tilde{\theta}(u) = a\cos\theta(u).$$

By substitution of this last equation in (5.7) we get, for all u in $U \bigcap \tilde{U}$,

$$p_x \cos\theta(u) - \tilde{p}_x a \cos\theta(u) = -p_y \epsilon \sqrt{1 - \cos^2\theta(u)} + \tilde{p}_y \tilde{\epsilon} \sqrt{1 - a^2\cos^2\theta(u)}.$$

with $\epsilon, \tilde{\epsilon} \in \{-1, 1\}$. This last equation implies that

$$P(X) = AX^2 - 2BX + C = 0, \tag{5.8}$$

with

$$\begin{aligned}
X &= \cos^2 \theta(u), \\
A &= \left((p_x - \tilde{p}_x a)^2 + \left(p_y - a^2 \tilde{p}_y\right)^2 \right) \left((p_x - \tilde{p}_x a)^2 + \left(p_y + a^2 \tilde{p}_y\right)^2 \right), \\
B &= \left(p_y^2 - \tilde{p}_y^2\right) \left(p_y^2 - a^2 \tilde{p}_y^2\right) + \left(p_y^2 + \tilde{p}_y^2\right) (p_x - \tilde{p}_x a)^2 , \\
C &= \left(p_y^2 - \tilde{p}_y^2\right)^2 .
\end{aligned}$$

Note that $p_y \neq 0$ by (5.4) and strict monotonicity. If one among the coefficients A, B, C is different from zero, this implies that X is a constant, *i.e.*, $\cos \theta(u)$ is a constant over $U \bigcap \tilde{U}$. Therefore $\theta(u)$ and $\theta(t)$ are constant over some nontrivial interval, this is impossible by strict monotonicity.

If A, B, C are all zero, this implies $p_y = \pm \tilde{p}_y$ by $C = 0$. Plugging in $B = 0$ gives $p_x - \tilde{p}_x a = 0$, since $p_y \neq 0$. Plugging both in $A = 0$ gives $a = \pm 1$.

Hence, either for some u_1, u_2 in $U \bigcap \tilde{U}$, $D(u_1, u_2) \neq 0$, and $p_x, \tilde{p}_x$ are uniquely determined or $\cos(\theta(u))$ and $\pm \cos(\tilde{\theta}(u))$ coincide on $U \bigcap \tilde{U}$. $\square$

By the previous lemma, given two monotonic compatible experiments (renormalized as above, *i.e.*, such that $q(0) = (0, 0, 0)$), there are two cases:

1. the constants $p_x, \tilde{p}_x$ are uniquely determined, or;
2. $\cos(\theta(u)) = \pm \cos(\tilde{\theta}(u))$ on some nontrivial interval. This implies that $\theta(u) = \pm \tilde{\theta}(u)$ (the situation $\theta(u) = \pi \pm \tilde{\theta}(u)$ does not happen since $\theta(0) = \tilde{\theta}(0) = 0$).

Case 1 implies that $L(\cdot)$ is uniquely determined by (5.6).

Case 2 implies that, after normalization $q(0) = (0, 0, 0)$ of the experiments, $u(t) = \tilde{u}(\pm t)$. This is equivalent to the fact that both experiments are either the same or deduced one from the other by an isometry of the plane, that changes orientation (see Rem. 5.5 below).

Definition 5.3. We say that two experiments are M-different, if their x, y trajectories cannot be deduced one from the other by an isometry of the plane.

We have proven the following theorem:

Theorem 5.4. Considering system (5.1), the problem $(\reflectbox{$\mathbf{P}$}_{\reflectbox{$\scriptstyle L$}})$ can be solved if the set of experiments is composed by two monotonic, M-different and compatible experiments which visit common values of the control.

Remark 5.5. One can read, in (5.6) that if $u(t)$ is changed for $u(-t), x(t)$ is changed for $-x(-t)$, $\theta(t)$ is changed for $-\theta(-t)$, then $L(u)$ remains unchanged. It means that we cannot expect to determine $L(\cdot)$ from two experiments deduced one from the other by a reflexion w.r.t. some axis. This corresponds also to the trajectory $(\tilde{x}(t), \tilde{y}(t))$ being the trajectory $(x(t), y(t))$ followed in reversed time.

5.3. Numerical results

The theory exposed in the preceding subsections was tested by means of both simulated and experimental data. More detailed simulation results will be presented in a forthcoming paper, relative to our global project on drones. Roughly speaking, the reconstruction algorithm is, as we explained in Section 3.3, based upon least-squares applied to equation (5.6), in order to reconstruct the adjoint vectors. Once the adjoint vectors are known, the cost can be reconstructed on the full domain visited by all experiments.

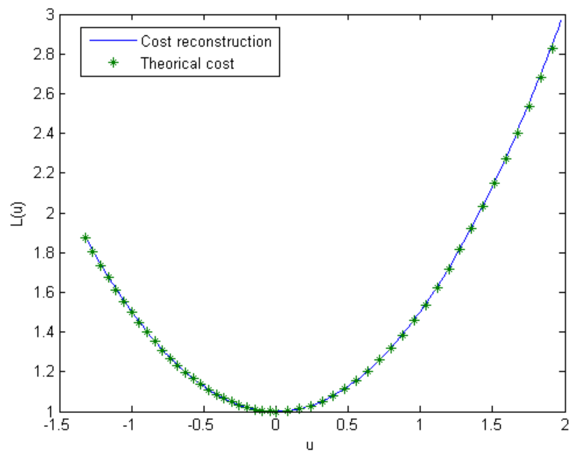


FIGURE 1. Ideal reconstruction

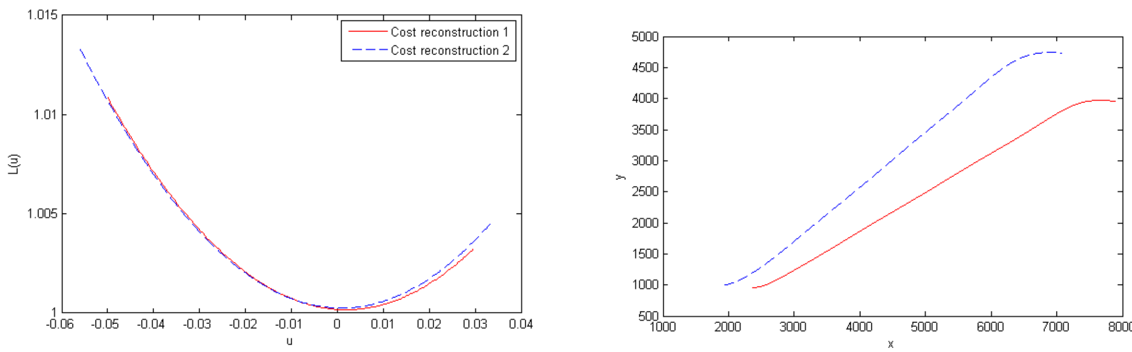


FIGURE 2. Reconstruction (left) on the basis of two experiments from a pilot (right).

Figure 1 shows just reconstruction based upon perfect experiments, computed from the known cost. Of course, this is just checking that the theory is not false, and that no purely numerical problem appears. This last point is not so obvious, as explained just below.

On the other hand, experimental data were obtained from the simulator (developed in our project) [2] in its manual mode. Datas shown here, in Figure 2 come from a beginner pilot. The pilot was just asked twice to change from flying direction for a parallel one, which implies the existence of an inflexion point (zero curvature). For the sake of testing the algorithm performances, we considered minimum possible level of information, *i.e.*, two trajectories only. Both trajectories were restricted to the maximum monotonicity interval.

One can see that, in this type of experiment (change for a parallel way), monotonicity looks not satisfied (it is very close not to be, and here, there is a real sensitivity problem in practice). Two costs have been reconstructed from the determination of both adjoint vectors, obtained from least squares.

One can see that this pilot actually looks to minimize a cost of the requested form, and this cost looks properly reconstructed (both reconstructions are very close).

Figure 3 shows a reconstruction, from the same pilot, on the basis on two trajectories without inflexion points. These second experiments have been performed one month later.

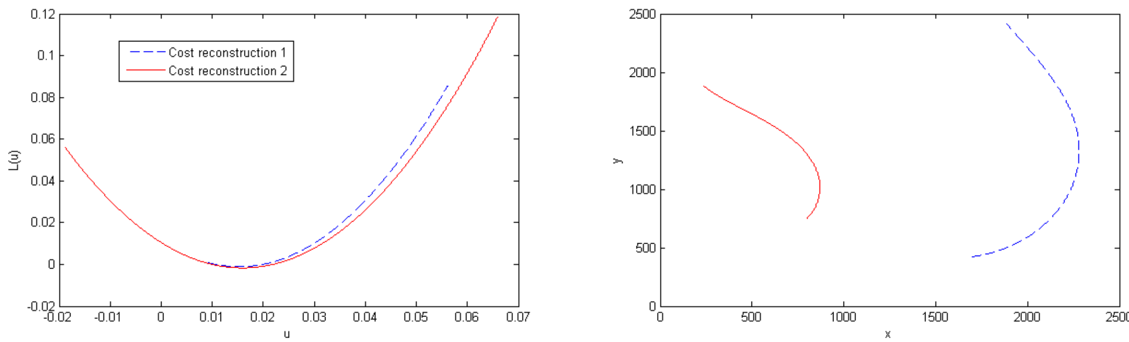


FIGURE 3. Non-inflexional experiments, same pilot.

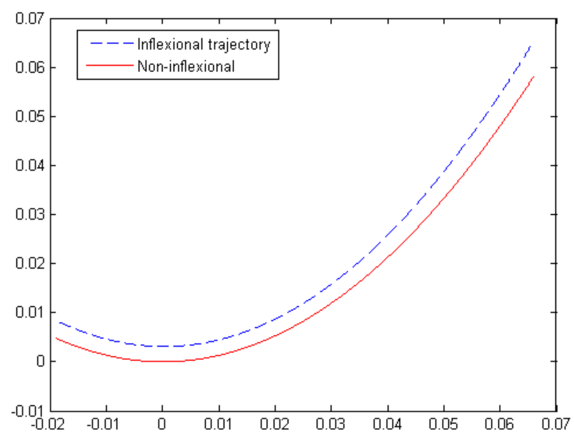


FIGURE 4. Comparison of both reconstructed criteria.

In Figure 4, we show a comparison of the reconstructed criteria for the two previous couples of experiments. Of course we make the best possible fitting using our correction term $L(u) = a(\tilde{L}(u) + bu)$. It seems that this pilot really minimized the same cost.

As a conclusion, the first trials we made seem to confirm the applicability of the method, and the reliability of the algorithm.

A measurement series with experimented pilots is on the point to be performed on their own training simulator.

6. Extension of the results to the case $\dot{q} = F_0(q) + uF_1(q)$

In this section, we generalize the results obtained in the previous sections to the more general class of smooth systems of the form

$$\dot{q} = F_0(q) + uF_1(q), \qquad q \in Q, \quad u \in \mathbb{R}, \tag{6.1}$$

with the state space Q being an n-dimensional connected, Hausdorff and paracompact smooth manifold.

To do so we follow exactly the same lines as in the previous sections and we do not repeat all the computations but we only point out the main differences. It is worth to notice that this case is even easier to handle.

In particular, the irrelevant linear degree of freedom in the cost $L(u)$ disappears: as explained below, the cost is now determined up to a multiplication by a positive scalar only and a simpler normalization is now given in Lemma 6.2.

As for system (2.1), we consider the problems $(\mathbf{P}_L)$ and $(\reflectbox{$\mathbf{P}_L$})$, but for the general system (6.1) now.

Similarly to Section 3.1.2, we define the Hamiltonian function $h(\lambda, p, q, u) = pF_0(q) + upF_1(q) + \lambda L(u)$, with $p \in \mathbb{R}^n$ and $\lambda \leqslant 0$, in order to apply the PMP to Problem $(\mathbf{P}_L)$.

Again, abnormal extremals have to be avoided, and we consider only normal extremals only, *i.e.*, we assume that $\lambda = -1$.

All the information is still contained in the system:

$$\begin{cases} \dot{q} = F_0(q) + uF_1(q) \\ \dot{p} = -p\dfrac{\partial F_0}{\partial q} - up\dfrac{\partial F_1}{\partial q} \\ L^*(pF_1(q)) + pF_0(q) = 0. \end{cases} \tag{6.2}$$

Using the u-parametrization of the experiments and taking into account the fact that p satisfies a linear ODE, the last equation of the previous system rewrites

$$L(u) - uL'(u) = p_0 V(u), \tag{6.3}$$

where p_0 is the initial value of $p(\cdot)$, and $V(u)$ is defined by

$$p_0 V(u) = p(t(u))F_0(q(t(u))).$$

Most of the material and results developed in the previous sections remain unchanged with system (1.3) replaced by system (6.1). The main change is that the criterion $L(u)$ is now reconstructed modulo multiplication by a scalar only. Indeed, the transformation $L \to aL(u) + bu$ does not leave invariant the set of extremal trajectories, only the transformation $L \to aL$ does. For this reason, Lemmas 3.14 and 3.15 modify as follows.

Lemma 6.1. *Let a be a positive number and let b be real. The two optimal control problems $(\mathbf{P}_L)$ and $(\mathbf{P}_{aL})$ have the same set of extremal trajectories, while, in general, the two optimal control problems $(\mathbf{P}_L)$ and $(\mathbf{P}_{L+bu})$ do not.*

Lemma 6.2. *At u_0, the cost function L to be reconstructed can be normalized so that $L''(u_0) = 1$.*

Besides this, Theorem 4.1 is also true for system (6.1) and its proof generalizes straightforwardly, with some extra work.

References

[1] A.A. Agrachev and Y.L. Sachkov, *Control theory from the geometric viewpoint.* Springer-Verlag, Berlin, *Encyclopaedia of Mathematical Sciences* **87** (2004). Control Theory and Optimization, II.

[2] A. Ajami, T. Maillot, N. Boizot, J.-F. Balmat, and J.-P. Gauthier. Simulation of a uav ground control station, in *Proceedings of the 9th International Conference of Modeling and Simulation, MOSIM'12* (2012). *To appear*, Bordeaux, France (2012).

[3] G. Arechavaleta, J.-P. Laumond, H. Hicheur, and A. Berthoz, Optimizing principles underlying the shape of trajectories in goal oriented locomotion for humans, in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on* (2006) 131–136.

[4] G. Arechavaleta, J.-P. Laumond, H. Hicheur, and A. Berthoz, On the nonholonomic nature of human locomotion. *Autonomous Robots* **25** 2008 25–35.

[5] G. Arechavaleta, J.-P. Laumond, H. Hicheur, and A. Berthoz, An optimality principle governing human walking. *Robot. IEEE Trans. on* **24** 2008) 5–14.

[6] B. Berret, C. Darlot, F. Jean, T. Pozzo, C. Papaxanthis, and J.P. Gauthier, The inactivation principle: mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements. *PLoS Comput. Biol.* **4** (2008) 25.

[7] B. Berret, J.-P. Gauthier, and C. Papaxanthis, How humans control arm movements. *Tr. Mat. Inst. Steklova* **261** (2008) 47–60.

[8] Y. Chitour, F. Jean, and P. Mason, Optimal control models of goal-oriented human locomotion. *SIAM J. Control Optim.* **50** (2012) 147–170.
[9] H. Chitsaz and S. LaValle, Time-optimal paths for a dubins airplane, in *Decision and Control, 2007 46th IEEE Conference on* (2007) 2379–2384.
[10] F. Chittaro, F. Jean, and P. Mason. On the inverse optimal control problems of the human locomotion: stability and robustness of the minimizers. *J. Math. Sci.* (*To appear*).
[11] J.-P. Gauthier, B. Berret, and F. Jean. A biomechanical inactivation principle. *Tr. Mat. Inst. Steklova* **268** (2010) 100–123.
[12] M. Golubitsky and V. Guillemin, *Stable mappings and their singularities.* Springer-Verlag, New York, Graduate Texts in Mathematics **14** (1973).
[13] F. Jean, Optimal control models of the goal-oriented human locomotion, Talk given at the "Workshop on Nonlinear Control and Singularities", Porquerolles, France (2010).
[14] W. Li, E. Todorov, and D. Liu, Inverse optimality design for biological movement systems. *World Congress* **18** (2011) 9662–9667.
[15] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, and E.F. Mishchenko, *The mathematical theory of optimal processes*, Translated from the Russian by K.N. Trirogoff, edited by L.W. Neustadt. Interscience Publishers John Wiley & Sons, Inc. New York-London (1962).
[16] S.M. Rump, Verification of positive definiteness. *BIT* **46** (2006) 433–452.
[17] R. Thom, Les singularités des applications différentiables. *Ann. Inst. Fourier Grenoble* **6** (1955–1956) 43–87.
[18] R. Vinter, *Optimal control*, Systems & Control: Foundations & Applications. Birkhäuser Boston Inc., Boston, MA (2000).

Chapitre 8

Conclusion

Le travail effectué dans cette partie concerne la recherche d'une méthode anthropomorphique pour améliorer les trajectoires obtenues en utilisant les algorithmes de la partie I.

En particulier, nous avons analysé le problème de contrôle optimal inverse associé aux drones.

Le chapitre 6 introduit quelques méthodes bio-inspirées utilisées pour appréhender le comportement d'animaux et définit le problème de contrôle optimal inverse visant à expliquer le comportement humain.

Nous avons analysé le problème de contrôle optimal inverse pour des pilotes dans le chapitre 7. Nous avons démontré que, pour un système général non-linéaire à un contrôle, il faut au moins trois expériences d'un même pilote pour reconstruire le coût minimisé. Dans le cas d'un drone de type HALE, le problème est légèrement différent et seulement deux expériences sont nécessaires à la reconstruction du coût minimisé.

Ces coûts reconstruits sont utilisables dans une méthode de planification optimale pour construire des trajectoires proches de celles produites par des pilotes expérimentés.

Conclusion générale

Dans ce travail, nous avons étudié différentes méthodes de planification pour drones, de type HALE et MALE.

La partie I nous a d'abord permis de faire l'état de l'art sur les méthodes de résolution actuelles. Nous avons étudié dans le chapitre 2 les modèles cinématiques des véhicules considérés. À la fin de ce chapitre, nous avons présenté différentes méthodes de planification point-point, dans un milieu encombré ou non.

De nouvelles méthodes de planification, permettant de répondre au problème de couplage vecteur/capteur, ont été proposées dans le chapitre 3. En particulier, nous avons suggéré une méthode quadratique permettant d'asservir de façon pondérée la trajectoire du couple drone/caméra.

Une de nos conclusions, dans le cas d'une caméra non contrainte, est que le problème de couplage vecteur/capteur se réduit au problème de planification du vecteur. Dans ce cas, une méthode indépendante de contrôle de la caméra a été proposée.

Le problème de planification point-pattern a ensuite été traité. Deux méthodes ont été étudiées, l'une basée sur le principe de LaSalle et l'autre sur le PMP.

Pour terminer, nous avons utilisé toutes les méthodes présentées pour proposer et appliquer un algorithme de suivi de cible mobile, en milieu contraint.

Dans la seconde partie, nous avons étudié une méthode anthropomorphique pour améliorer les algorithmes de planification exposés dans la partie I.

En particulier, nous nous sommes intéressés à un problème de contrôle optimal inverse et avons obtenu des résultats nouveaux, théoriques mais constructifs :

- Au chapitre 6, nous avons introduit les méthodes bio-inspirées et le problème de contrôle optimal inverse.
- Une méthode d'estimation du coût optimisé par les pilotes a été détaillée au chapitre 7. Nous avons prouvé que, pour un système général non-linéaire à un contrôle, il faut avoir au moins trois expériences d'un même pilote pour reconstruire le coût minimisé. Dans le cas d'un drone de type HALE, le problème est légèrement différent et seulement deux expériences sont nécessaires à la reconstruction du coût minimisé.

Quelques perspectives pour continuer ce travail sont listées ci-dessous.

Appliquer les méthodes de planification sur un drone

Les méthodes étudiées au cours de ce travail ont été testées en simulation uniquement. Il serait intéressant de les tester dans un cadre plus réaliste.

Réaliser la synthèse optimale en temps pour n'importe quel type de pattern

La contribution que nous avons apportée concerne, entre autres, certaines méthodes de planification vers un cercle de rayon le rayon de courbure minimal du vecteur. Il serait intéressant d'étudier ces méthodes de planification pour tous types de patterns.

Étudier les méthodes de localisation de la cible

Dans ce travail, la position de la cible est supposée connue à chaque instant. Cette hypothèse n'est pas toujours satisfaite. Il faut donc mettre en place des algorithmes robustes de traitement d'images (ou toute autre méthode permettant de gérer des données capteurs) pour déterminer la position de la cible.

Étudier le problème de contrôle optimal inverse associé aux drones MALE

La partie II présente des résultats concernant l'apprentissage à partir de données humaines. Nous avons considéré que les expériences faites vérifient un système de type Dubins. Cette hypothèse est valable uniquement si le drone étudié est de type HALE. Nous pourrions nous intéresser aux drones de type MALE. Pour cela, il faudrait étudier le problème de contrôle optimal inverse associé à un système différentiel non-linéaire à plusieurs contrôles.

Étudier le problème de contrôle optimal inverse pour les drones HALE et MALE dans le cas où le pilote doit rejoindre un pattern

Le problème de contrôle optimal inverse étudié correspond à des expériences de planification point-point. Or, nous avons vu que le problème de planification point-pattern est important. Une idée serait alors de considérer le problème inverse associé au problème point-pattern qui ajouterait une condition de transversalité sur la cible associée au problème de contrôle optimal.

Annexes

Annexe A

Article de conférence concernant le simulateur [6] : *"Simulation of a UAV ground control station"*

L'article présenté a été accepté à la conférence *MOSIM'12 :* 9th *International Conference of Modeling, Optimization and Simulation* [6].

9^{th} *International Conference of Modeling, Optimization and Simulation - MOSIM'12*
June 6-8, 2012 - Bordeaux - France
"Performance, interoperability and safety for sustainable development"

SIMULATION OF A UAV GROUND CONTROL STATION

A. AJAMI, T. MAILLOT, N. BOIZOT, J-F. BALMAT

LSIS, Université du Sud Toulon Var
Avenue de l'Université
83957 LA GARDE CEDEX - FRANCE
alain.ajami@univ-tln.fr, thibault.maillot@univ-tln.fr
nicolas.boizot@univ-tln.fr, balmat@univ-tln.fr

J-P. GAUTHIER

LSIS, Université du Sud Toulon Var
and
INRIA TEAM "GECO"
gauthier@univ-tln.fr

ABSTRACT: *In this article we present the development of a UAV ground control station simulator. We propose a module based description of the architecture of this simulator. We recall the nonlinear model of a fixed wing aircraft. Finally we outline ideas for improved planification tasks. The approach is made clear through several diagrams, figures of the resulting station are displayed.*

KEYWORDS: *planification, simulation, UAV, optimal control, inverse optimal control problem*

1 INTRODUCTION

Today's interest in drone technology led to an increasing number of dedicated projects (Tisdale, Kim & Hedric 2009, Kim, Shim & Sastry 2002, Fabiani, Fuertes, Piquereau, Mampey & Teichteil-Königsbuch 2007, Ippolito, Yeh & Campbell 2009). Those projects tackle problems such as the autonomy improvement, the reduction of drone crashes due to poor availability of information, the organization of drone swarms or the calculation of secure and optimal flight paths. LSIS laboratory joins this research effort in the framework of project SHARE. Supported by a consortium of companies and research labs[1], the research aims to develop a universal and interoperable ground control station for fixed and rotary wing UAVs with a reduced number of operators. Specifically LSIS lab focuses on the connections between the UAV trajectory and its sensors. As a consequence, improved path planning algorithms that take into account payload requirements, optimal costs and obstacles (or no flight zones) avoidance are needed.

In order to test and demonstrate our techniques a ground control station simulator is required. The use of engineering and simulation softwares allows to shorten development time and still maintain portability of the code. Since the development is done in close contact with research partners, modularity is a key factor. Indeed, we want to be able to easily add, remove or modify parts of the simulator. Another challenge is to determine the automation level of the station. (Cummings, Platts & Sulmistras 2006, Sheridan, Verplank & Brooks 1978) propose insights on automation strategies that are useful to describe the simulator. On the Sheridan-Verplank scale, the station ranks no more than 3 and, according to (Cummings et al. 2006), it has a level of interoperability of 4 with respect to the STANAG 4586 classification. This latter level can be described as follows: *the ground station allows the control and monitoring of the UAV at the exception of launch and recovery situations*. The simulation of such a device starts with the simulation of the aircraft's trajectory. Then follows the emulation of all the data exchanged through the station between the operator and the UAV, and finally of the control algorithms.

Classic tools for the implementation of the aircraft's dynamics, the controller part and the man machine interface are Matlab and Matlab/Simulink. We prefered the use of S-functions to a systematic block based implementation in order to ease portability in case of discard of Matlab's automatic code generation features. The simulation of the virtual environment and of the video flow is performed by a flight simulator (Craighead, Murphy, Burke & Goldiez 2007). Flightgear which is open source and interfaces nicely with Matlab/Simulink is used (jun Yang, hui Qi & lin Shan 2009, Sorton & Hammaker 2005).

The rest of this article is organised as follows. The structure of the simulator is presented in section 2. It is broken into several modules which functionalities are explained. The aircraft dynamics are recalled in section 3. Although project SHARE aims at a ground control station that addresses both fixed and

[1] Opéra Ergonomie, ONERA, Thales Alénia Space, Eurocopter, Adetel group.

rotative wing aircrafts, we focus on fixed wing ones. Finally section 4 contains both low and high level descriptions of the control module (cf. figure 4). The reader is advised to reserve a special attention to subsection 4.2 where important ideas regarding high level planification and learning algorithms are sketched.

2 SIMULATOR

A ground control station simulation instance has two main parts: the initialization phase and the scenario realization. The initialization phase allows to define parameters that are specific to a given simulation instance, this step is detailed in subsection 2.1. During simulation several different missions can be realized accordingly to the operator's scenario. The architecture of this scenario realization part is detailed in subsection 2.2.

2.1 Simulator's Initialization Phase

We divide simulation parameters into two categories: setup parameters and core parameters. Core parameters contribute to the simulation inner mechanisms: model parameters (subsections 3.3.1, 3.3.2 and 3.4) and control parameters (section 4). As such the user cannot manipulate them directly: they are either loaded through a *setup assistant* utility or managed through a *modify/create assistant*.

The initialization procedure is schematized in figure 1 below. The user selects a *UAV type* between fixed and rotary wings aircrafts which determines the equations of motion that are solved. The *model choice* allows to specify which particular aircraft is simulated. The aerodynamic model (subsection 3.3.2), the corresponding aerodynamic parameters and the level one controller coefficients are loaded accordingly. Those can be further modified by the *level one controls efficiency* parameter (i.e. regular or degraded mode).

Several *payloads* are embedded on the UAV. They are selected out of a list. In particular camera parameters are specified (subsection 3.4). Noise levels and bias of the sensors are also defined. High level planification algorithms *constraints* are specified in the form of minimum/maximum time to next waypoint, energy consumption related constraints, stealth mode, line of sight preservation or, more technically, weighting parameters for the algorithms cost functions.

Finally the set of *maps* of the area where the scenario takes place and the *initial position of the aircraft*[2] are selected. At the end of the initialization phase a *map synchronization utility* ensures the connection between the 3D virtual environment and the 2D maps of the area of interest.

[2]i.e. the origin of the world frame defined in subsection 3.2

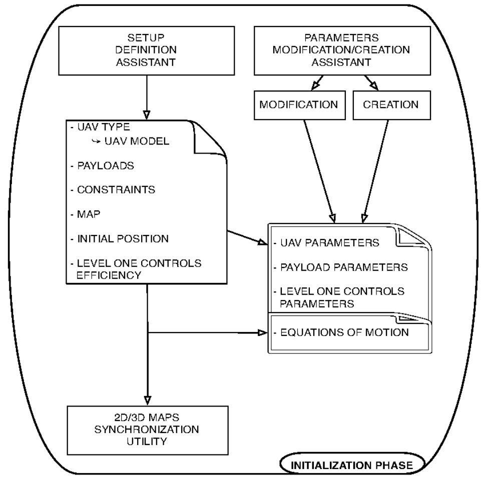


Figure 1: Functional diagram of the initialization phase

2.2 The Scenario

In this subsection we propose a module based presentation of the simulator. This approach is schematized in figure 2.

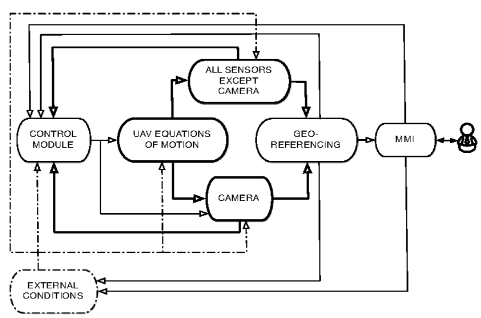


Figure 2: Module based architecture diagram

Let us start with the resolution of the *UAV equations of motion*. As said before they are completely determined by the choices made in the initialization phase. Input and output vectors are defined in subsection 3.1 for a fixed wing aircraft. Let us focus on the output of this module, namely the state vector. It is used to generate the sensors outputs which are lumped into two categories: *all sensors except the camera* and *the camera*. The state vector is actually fully available from the several sensors a UAV carries. However we want to be able to simulate desynchronized sensors, noise levels and bias. The position which is available in the (x, y, z) system of coordinates is transformed into the WGS84 system for geo-localization purposes (Grewal, Weill & Andrews 2007). The UAV's attitude transforms from the unit quaternion representation to the Roll-Pitch-Yaw angles one. The camera has a special treatment since first, a set of differential equations has to be solved and second, the video flow is not generated in this module. We actually only take care of the line of sight orientation calculation and the evolution of extra camera parameters if they are used (subsection 3.4).

The outputs of the two sensors related modules are sent to the *geo-referencing module*. This module ensures the synchronization between the 2D map, the 3D environment and the radar representation. The 2D synchronization allows to display the trajectory of the aircraft on a map, to represent the video cone and the targets whose positions are known to the ground station. The radar gives information of the relative positions of possible targets. Finally the 3D synchronization is the computation of the actual video flow (i.e. Flightgear). Those data are sent to the *man-machine interface* (MMI) module for display (figure 3). This module is detailed further below in the present subsection.

All the modules described before feed the *control module* which in return sends its data to the *equations of motion* and *camera* modules. It is the object of section 4.

The *exterior conditions* module is the last part of this diagram, it encapsulates all the other ones since perturbations can be introduced in all the other blocks. At this stage of the development we only use it to manage weather conditions, more particularly wind. A Von Karman model is used to generate the wind velocity vector required by the resolution of the equations of motion. In our opinion, at least two extra modules shall be considered. We didn't add them in the diagram for clarity purposes and since they are at early development stages. Those are the *extra payloads management* (i.e. sensors specific to a mission, packages needed to be dropped...) and the *targets management unit*.

We end this subsection with a few words on the *man-machine interface* module. Figure 3 gives an account of how data flows from the operator to the UAV and back. Whereas the information sent back to the operator talk for themselves, it is not the case of all the features managed by the operator. *UAV and payloads control modes* are given in the next section. *Simulation management features* are start, pause, stop and visualization options. As for the rest, it is pretty obvious except for the *mission type management* part. This task is performed by the MMI modules which triggers the adequate module functionalities accordingly to the operator's will. Let us focus on possible missions. The UAV has to perform several types of mission (NATO 2008) of which we implemented the following ones.

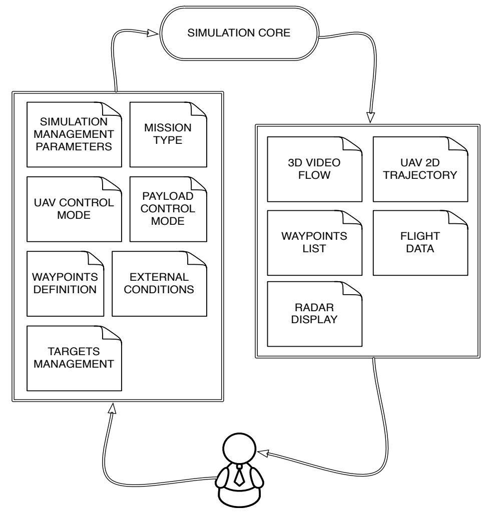


Figure 3: Man-machine interface data flow diagram

Follow flight plan This is the basic mission. A flight plan is a series of waypoints the UAV has to visit. The flight plan is either defined before the scenario or made by the operator as the scenario undergoes.

Patterns This mission consists in following operator parameterized patterns: circle, racetrack, and figure eight (table 1) in order to detect any activity in a specific area and/or collect information. The trajectory that leads to a given pattern is the subject of more or less sophisticated algorithms (section 4). Indeed the UAV has to be tangent when it enters the pattern. The approach trajectory to the pattern is constrained by the geographical configuration, UAV physical characteristics, UAV inital orientation and the traveling direction of the pattern.

Tracking The objective is to follow a friend or enemy point of interest. In the case of an ally, an expanded surveillance zone is considered. Multiple situations can be encountered: the target is in a no-fly zone, target occlusion occurs, the target is moving, camera commands are either automated or manually activated by the operator. In the case of an enemy target, it is of utmost importance to be able to maintain it in the camera field of view.

Replanification Replanification happens when the UAV needs a waypoints list that is better generated with appropriate algorithms – i.e. when a mission has to be changed because of unexpected events. No-fly zone avoidance, search of an admissible path and optimization of an optimality criteria are the three aspects that drive this mission type (subsection 4.2).

pattern	parameters list
common to all	center point rotation direction
circle	radius
racetrack / hippodrome	small radius large radius orientation
figure eight	pseudo radius orientation

Table 1: Pattern parameterization

3 MODELING

3.1 Notations and Definitions

$v^{(f)}$	vector v projected in frame (f)
Id_n	$(n \times n)$ dimensional identity matrix
$X^{(f)}$	aircraft's position vector
$V^{(f)}$	aircraft's speed vector
$V_{ae}^{(f)}$	aerodynamic speed vector
q	unit attitude quaternion
$\left[(\tilde{q})^{\times}\right]$	matrix associated to $v \mapsto \tilde{q} \times v$
$\phi^{(w)}$ (resp. ϕ_c)	aircraft's (resp. camera's) roll angle
$\theta^{(w)}$ (resp. θ_c)	aircraft's (resp. camera's) pitch angle
$\psi^{(w)}$ (resp. ψ_c)	aircraft's (resp. camera's) yaw angle
$\omega^{(f)}$	aircraft's angular velocity vector
$(x_c, y_c, z_c)^{(w)}$	camera's position
α	aerodynamic angle of attack
β	aerodynamic sideslip angle
$F_{ae}^{(b)}$	aerodynamic force
$F_{pr}^{(b)}$	propulsive force
$M_{ae,g}^{(b)}$	aerodynamic moment at point g
$M_{pr,g}^{(b)}$	propulsive moment at point g
$G^{(f)}$	gravity vector (i.e. $G^{(w)} = (0, 0, g)$)
I	aircraft's inertia matrix
δ_l	roll control
δ_m	pitch control
δ_n	yaw control
δ_x	thrust control
T	overall thrust
δ_ϕ	$\phi_c^{(b)}$ set point
δ_θ	$\theta_c^{(b)}$ set point
δ_ψ	$\psi_c^{(b)}$ set point
$u(t)$ (resp. $u_c(t)$)	UAV (resp. camera) control vectors
ρ	atmospheric density
ρ_0	atmospheric density at sea level
S	aircraft's surface of reference
l	aircraft's length of reference
m	mass of the aircraft

The controls are

$$u(t) = (\delta_l(t), \delta_m(t), \delta_n(t), T(t))^T$$
$$u_c(t) = (\delta_\phi(t), \delta_\theta(t), \delta_\psi(t))^T$$

and the state vector is

$$\left\{\begin{array}{rcl} X^{(w)} & = & \left(X_x^{(w)}, X_y^{(w)}, X_z^{(w)}\right)^T \\ V^{(b)} & = & \left(V_x^{(b)}, V_y^{(b)}, V_z^{(b)}\right)^T \\ q & = & (q_0, \tilde{q})^T = (q_0, q_1, q_2, q_3)^T \\ \omega^{(b)} & = & \left(\omega_1^{(b)}, \omega_2^{(b)}, \omega_3^{(b)}\right)^T \end{array}\right.$$

The matrices $\left[(\tilde{q})^{\times}\right]$ and I are defined as

$$\left[(\tilde{q})^{\times}\right] = \begin{pmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{pmatrix}$$

and $I = I^T = \left(I_{i,j} \in \{1,2,3\}^2\right)$

3.2 Hypotheses and Frames

The model for aircraft dynamics proposed here is inspired by (Boiffier 1998, Wanner 1984, Junkins & Schaub 2001). The modeling hypotheses are[3]

1. earth is flat and motionless with respect to the simulation duration,
2. gravity is constant (i.e. the center of mass is the same as the center of gravity),
3. atmospheric pressure and temperature are constant (i.e. only density variation with respect to height is considered),
4. there is a uniform wind velocity field,
5. the aircraft is a 6 DoF solid which fuselage has a symmetry plane,
6. the propulsion system is on the fuselage axis,
7. the mass and the inertia matrix of the aircraft are constant parameters.

The first hypothesis makes the world frame inertial, thus we only need three frames to derive the equations.

World frame (w) This frame is attached to a reference point O, the Ox, Oy, Oz axes are oriented in the north-east-down directions. Although the aircraft's position expressed through the equations of motion is given in the (x, y, z) system, we need to express it in the WGS84 system in order to perform geo-referencing (Grewal et al. 2007).

[3] See (Boiffier 1998) for a comprehensive treatment of the hypotheses.

Body frame (b) This frame is attached to the aircraft's center of mass (G). The $Gx^{(b)}$ axis goes along the fuselage axis. The $Gz^{(b)}$ axis is taken in the plane of symmetry of the aircraft and points downward, and $Oy^{(b)} = Oz^{(b)} \times Ox^{(b)}$
The composition of a translation and a rotation transforms frame (w) into frame (b). The rotation defines the attitude of the aircraft. We use the Euler angles *Roll-Pitch-Yaw* parameterization for its physical meaning, and the unit quaternion one to solve differential equations (Junkins & Schaub 2001).

Aerodynamic frame (a) This frame shares his origin with frame (b). The $Gx^{(a)}$ axis is carried by the aerodynamic velocity vector. The rotation that transforms $Gx^{(b)}$ into $Gx^{(a)}$ is parameterized by the angle of attack and the sideslip angle. The other axes are obtained through this rotation (subsection 3.3.2).

3.3 Aircraft Dynamics

3.3.1 Equations of Motion

The dynamic resultant theorem and the dynamic moment theorem write as

$$m\frac{d}{dt}\dot{V}^{(b)} = F_{ae}^{(b)} + F_{pr}^{(b)} + mG^{(b)} - \omega^{(b)} \times mV^{(b)} \tag{1}$$

$$I\dot{\omega}^{(b)} = M_{ae,g}^{(b)} + M_{pr,g}^{(b)} - \omega^{(b)} \times I\omega^{(b)} \tag{2}$$

where

$$F_{ae}^{(b)} = \frac{\rho S V_{ae}^2}{2} \begin{pmatrix} -C_A^{(b)} \\ C_Y^{(b)} \\ -C_N^{(b)} \end{pmatrix} \qquad F_{pr}^{(b)} = \begin{pmatrix} F_{pr,x}^{(b)} \\ F_{pr,y}^{(b)} \\ F_{pr,z}^{(b)} \end{pmatrix} \tag{3}$$

$$M_{ae}^{(b)} = \frac{\rho S l V_{ae}^2}{2} \begin{pmatrix} C_l \\ C_m \\ C_n \end{pmatrix} \qquad M_{pr}^{(b)} = \begin{pmatrix} M_x^{(b)} \\ M_y^{(b)} \\ M_z^{(b)} \end{pmatrix} \tag{4}$$

The rotation matrix from the world frame to the body frame expressed in terms of q, and the dynamics of q are (Junkins & Schaub 2001)

$$R(q) = \left(q_0^2 - \tilde{q}\tilde{q}^T\right) Id_3 + 2\left(\tilde{q}^T\tilde{q} - q_0\left[\tilde{q}^\times\right]\right) \tag{5}$$

and

$$\dot{q} = \frac{1}{2}\begin{pmatrix} 0 & -\omega^{(b)} \\ \omega^{(b)} & -\left[\left(\omega^{(b)}\right)^\times\right] \end{pmatrix} q \tag{6}$$

Therefore we have

$$mG^{(b)} = R(q)\,(0,0,mg)^T \tag{7}$$

$$\dot{X}^{(w)} = V^{(w)} = R^T(q)V^{(b)} \tag{8}$$

The equations of motion of the aircraft with state $\left(X^{(w)}, V^{(b)}, q, \omega^{(b)}\right)$ are equations (8), (1), (6) and (2). In order to actually use them, we still need to define a few elements.

- The atmospheric density model is

$$\rho = \rho_0 e^{-1.1210^{-4}h} = \rho_0 e^{1.1210^{-4}z}$$

- From hypothesis (5), elements $I_{1,2}$ and $I_{2,3}$ of the inertial matrix are null (the symmetry of the inertial matrix implies $I_{2,1} = I_{3,2} = 0$).

- From hypothesis (6), the propulsion force vector components $F_{pr,y}^{(b)}$ and $F_{pr,z}^{(b)}$ are null. The propulsion moment vanishes – see (4.3.4) in (Boiffier 1998), with $\alpha_m = \beta_m = 0$.
  An engine efficiency model can be used for the modulus of the thrust (i.e. $F_{pr,x}^{(b)}$ in our case). For example, in (Boiffier 1998) $T = k_m \rho V_{ae}^\lambda \delta_x$ is proposed. We simply used $T = k_m\delta_x$, with $0 < k_m \leq 1$.

- The wind is characterized by its velocity vector $V_w^{(b)}$. The aerodynamic speed vector is then defined as $V_{ae}^{(b)} = V_w^{(b)} - V^{(b)}$, and $V_{ae}^2 = \left(V_{ae}^{(b)}\right)^T V_{ae}^{(b)}$

3.3.2 Aerodynamic Model

We call aerodynamic model the equations used to fully express aerodynamic forces and moments (equation 4). Force coefficients are first defined in frame (a) (i.e. from wind tunnel experiments), then rotated into frame (b). As such the angle of attack and the sideslip angle are needed. Starting from the aerodynamic speed vector we derive equations

$$\begin{aligned} V_{ae,x}^{(b)} &= V_{ae}\cos(\alpha)\cos(\beta) \\ V_{ae,y}^{(b)} &= V_{ae}\sin(\beta) \\ V_{ae,z}^{(b)} &= V_{ae}\sin(\alpha)\cos(\beta) \end{aligned}$$

and we write

$$\begin{pmatrix} C_A^{(b)} \\ \\ C_Y^{(b)} \\ C_N^{(b)} \\ \\ \end{pmatrix} = \begin{pmatrix} \mathbf{C_x^{(a)}}\cos(\alpha)\cos(\beta) + \mathbf{C_y^{(a)}}\sin(\beta)\cos(\alpha) \\ \qquad\qquad -\mathbf{C_z^{(a)}}\sin(\alpha) \\ \mathbf{C_y^{(a)}}\cos(\beta) - \mathbf{C_x^{(a)}}\sin(\beta) \\ \mathbf{C_x^{(a)}}\sin(\alpha)\cos(\beta) + \mathbf{C_y^{(a)}}\sin(\beta)\sin(\alpha) \\ \qquad\qquad +\mathbf{C_z^{(a)}}\cos(\alpha) \end{pmatrix}$$

There are many possible models for $C_x^{(a)}$, $C_y^{(a)}$, $C_z^{(a)}$, C_l, C_m and C_n (Wanner 1984, Schmidt 1998). We cite the simple model

$$\begin{array}{lcl} C_x^{(a)} & = & C_{x,0} + k_i C_z^2 \\ C_y^{(a)} & = & C_{y,\beta}\beta + C_{y,\delta_n}\delta_n \\ C_z^{(a)} & = & C_{z,\alpha}\alpha + C_{z,\delta_m}\delta_m \\ C_l & = & C_{l,\beta}\sin(\beta) + l\left(C_{l,\omega_1}\frac{\omega_1}{V_{ae}} + C_{l,\omega_3}\frac{\omega_3}{V_{ae}}\right) \\ & & \qquad + C_{l,\delta_l}\delta_l + C_{l,\delta_n}\delta_l \\ C_m & = & C_{m,\alpha} + C_{m,\delta_m}\delta_m + C_{m,\omega_2}\frac{l\omega_2}{V_{ae}} \\ C_n & = & C_{n,\beta} + C_{n,\delta_n}\delta_n + C_{n,\omega_3}\frac{\omega_3}{V_{ae}} \end{array}$$

3.4 Camera Dynamics

This model tracks the orientation of the line of sight of the camera. This latter is assumed to be solidly attached to the center of mass of the aircraft:

$$x_c^w = X_x^{(w)}$$
$$y_c^w = X_y^{(w)}$$
$$z_c^w = X_z^{(w)}$$

Following camera datasheet specifications (wescam 2011) the dynamics can be approximated as a first order response to an attitude set point. This attitude set point is given in the (b) frame (i.e. $\dot{\phi}_c^{(b)} = (-\phi_c^{(b)} + \phi_{set}^{(b)})/\tau$). The time constant τ defines the speed of the camera steering system. Constraints on the coverage can be expressed in terms of constraints on the inputs rather than on the output (a significant asset both for implementation and optimal control purposes).

$$\left\{\begin{array}{lcl} \dot{\phi}_c^{(w)} & = & \omega_1^{(b)} - \frac{1}{\tau}\left(\phi_c^{(w)} - \phi^{(w)}\right) + \frac{1}{\tau}\delta_\phi \\ \dot{\theta}_c^{(w)} & = & \omega_2^{(b)} - \frac{1}{\tau}\left(\theta_c^{(w)} - \theta^{(w)}\right) + \frac{1}{\tau}\delta_\theta \\ \dot{\psi}_c^{(w)} & = & \omega_3^{(b)} - \frac{1}{\tau}\left(\psi_c^{(w)} - \psi^{(w)}\right) + \frac{1}{\tau}\delta_\psi \end{array}\right. \quad (9)$$

Extra parameters that can used to model the camera system are described in the following table (wescam 2011).

parameter	indicative value
optical eye field of view	36 to 1 (°)
IR fields of view	30, 7 and 1.8 (°)
frequency	24 (FPS)
turret coverage	
azimuth	0 to 360 (°)
elevation	90 to -120(°)
roll	na
turret steering speed	0 to 60 (°/sec)
turret stabilization quality	abt 3.10^{-3} (°)
autofocus time constant	0
	(i.e. instantaneous)

4 CONTROL MODULE

As explained in (Cummings et al. 2006) both the level decomposition and strategies of automation are crucial elements in order to achieve efficient accurate and safe UAV operations. First we give the global picture of the control module in subsection 4.1, and then focus on two high-level aspects in subsection 4.2. Those two important topics are planification and learning techniques.

4.1 General Considerations

Three levels are considered for this module (figure 4). Level zero corresponds to a direct access to the control surfaces of the UAV. Although unrealistic we kept the possibility to directly manipulate control surfaces as far as the simulator only is concerned. Level one takes an input signal of the form *heading, speed, altitude* and outputs a $(\delta_l, \delta_m, \delta_n, \delta_x)$ vector (section 3). This controller consists in a stability augmentation system (yaw, pitch, and phugoid dampers) in series with a basic PID like autopilot system (heading, airspeed and altitude holders). Note that the complexity of this controller grows with the complexity of the aerodynamic model (recall that the controller parameters are loaded accordingly to the UAV model, cf. section 2). Finally level two encapsulates all the algorithms that generate a *heading, speed, altitude* set point.

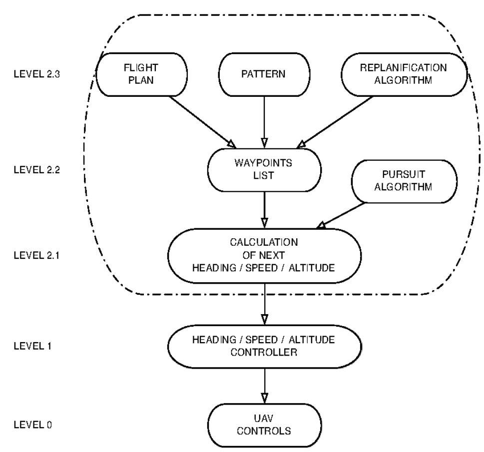


Figure 4: Level based decomposition of the controller module

The operator has the choice between several different piloting modes (figure 3) for both the UAV and the camera turret.

Manual mode Through a joystick, the *heading, speed, altitude* setpoint is directly controlled. As mentioned before, a very special option, only available in simulation, grants access to the control surfaces of the UAV.

Semi-manual mode At least one of the three components of the *heading, speed, altitude* setpoint is managed by the system – e.g. altitude is kept fixed.

Automatic mode The level one set point is defined by a level two algorithm which depends on the active mission.

The control modes of the camera are simpler to handle as only manual and automatic modes are proposed. Knowing the lastly visited, the targeted and the next on the list waypoints, the calculation of the new *heading, speed, altitude* set point is easily obtained from geometry. Note that a waypoint is defined by its WGS84 coordinates and an optional timestamp. When minimum/maximum time between waypoints constraints are activated, timestamps are used to adjust the set speed.

The next three parts to explain are the *pattern, replanification and pursuit* algorithms. As said before a pattern is defined by a set of parameters (table 1) as such a waypoints list describes it. What remains to be done is to propose a trajectory that allows the UAV to reach the pattern with the correct configuration (i.e. tangentially), starting from any initial point and any initial orientation. This is one of the jobs done by the *replanification* algorithm. The second one is to generate optimal trajectory to travel through an area without having a pre-established flight plan. A strategy in order to use this algorithm is to solve a replanification problem, then propose the solution trajectory as a list of waypoints and finally refresh the list through a new calculation each time a waypoint is reached. The *pursuit algorithm* output can be seen as a *heading, altitude, speed* vector that is recomputed whenever necessary.

A few insights on replanification algorithms are proposed in the next subsection. We even make one step farther by presenting experiment based learning of optimal cost functions.

4.2 Planification and Learning

4.2.1 Introduction

In the context of the SHARE project we have also to achieve two extra tasks:

1. Fill in the module called *planification algorithms* in figure 4. There is a lot of bibliography on this topic (Betts 2001, Bullo & Lewis 2004, Fliess, Lévine, Martin & Rouchon 1995, Kim et al. 2002, Laumond 1998, LaValle 2006, Park, Deyst & How 2004, Van Nieuwstadt & Murray 1998) , non exhaustively. These classical algorithms are based upon very different approaches such as geometric control, optimal control, flatness. In fact we are also developing our own methods that we present briefly below (subsection 4.2.2).
2. An idea to develop planification/replanification methods is to learn from the behavior of experimented pilots. We do this here for HALE (High Altitude Long Endurance) drones. In subsection 4.2.3 we present briefly our ideas, that are inspired from the beautiful work of Jean and al. in the papers (Chitour, Jean & Mason to be published, Chittaro, Jean & Mason to be published). In these papers, they attack the problem of identification of the cost minimized in human locomotion. Our problem is very similar since HALE drones behave kinematically more or less as a human being moving on a plane (constant altitude, constant speed). There is a lot of other methods dedicated to this human locomotion problem, see for instance (Li, Todorov & Liu 2011, Berret, Darlot, Jean, Pozzo, Papaxanthis & Gauthier 2008).

4.2.2 Planification/Replanification

In general the mission of a drone is planed in advance and specified by a certain number of checkpoints and a certain number of patterns (line, circle, figure eight and hippodrome). There is the need of on-line replanification methods when the mission is interrupted and the drone has to:

- Join a fixed target and turn around following a certain pattern,
- Join a moving target and follow it. This is not an obvious problem when the minimum speed of the drone is higher than the target speed.

Regarding the case (not yet treated above) of HALE rotorcraft-based drones, they behave kinematically more or less as the classical "simple car" model (Laumond 1998). Hence one can formulate an optimal control problem which looks like a left-invariant subriemannian problem over the group of motions of the plane. On this topic there is the beautiful complete mathematical work of Yuri Sachkov (Moiseev & Sachkov 2010), Sachkov 2010, Sachkov 2011) that does the job, if no obstacles are taken into account. If obstacles occur, the Sachkov method can be coupled to a method for finding first admissible trajectories avoiding obstacles and approximating them by pieces from the Sachkov synthesis. Several methods are available to find such admissible trajectories see (LaValle 2006).

The case of fixed wings drones is different. If we consider a simple model of Dubins type (Laumond 1998), the problem can be stated as a repetition of minimum-time problems for the Dubins car, where the final target is a (non-oriented or oriented) pattern. This optimal control problem can be solved easily, but it leads to a non-smooth and eventually non-continuous optimal synthesis. In fact, a nice smooth optimal synthesis can be found, leading to trajectories shown in figure 5. Details on this optimal synthesis and proof of the stability and convergence will be given in a forthcoming paper.

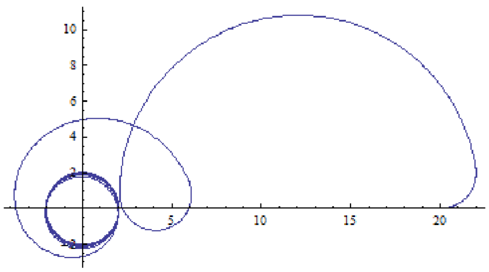


Figure 5: Trajectory resulting of smooth optimal synthesis starting at the point (20;0) with a direction of 0 radian and arriving tangent to a pattern, here the circle centering at (0;0)

4.2.3 Learning from Experimented Pilots

When the planning problems are specified in terms of optimal control problems, the choice of the cost to be minimized can be made from the experience of experimented pilots. For our fixed wings HALE drones, as we said, the basic kinematic model is the classical Dubins one. It has been shown by Jean and al. (Chitour et al. to be published, Chittaro et al. to be published) that the cost to be minimized for human locomotion is an integral length-curvature compromise. We completed their work by developing a program allowing to identify this cost. Moreover, we proved the following general theoretical result: for a generic model of motion, the integral cost can be recovered from two experiments, provided that these experiments visit at least twice the same value of the control. These developments will be the purpose of another paper. On the figure 6, we show the reconstruction of the cost as a function of length and curvature from two experiments.

5 CONCLUSION

In this article we presented a ground control station simulator for fixed wing aircrafts that can easily be extended to rotative wing ones. This description has been done in the form of a module based decomposition. The control module was given special attention as it deals with several issues important in order to improve the UAV autonomy. We sketched ideas to tackle the planification/re-planication task which can be used to deal with pattern based navigation and tracking problems. We also presented an approach to improve the cost functions that are used in optimal control synthesis based on trajectories obtained from piloted aircrafts.

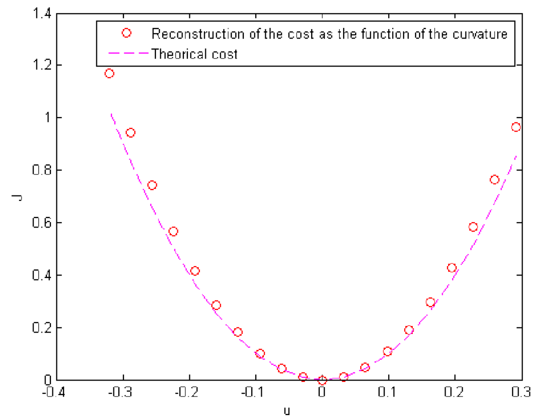


Figure 6: Reconstruction of the cost as the function of the curvature and the theorical cost

To conclude this work we display in figure 8 a view of the ground control station simulator. Snapshots of a monitoring mission of a famous building while flying in circles are shown in figure 7 (white squares have been added to ease visualization).

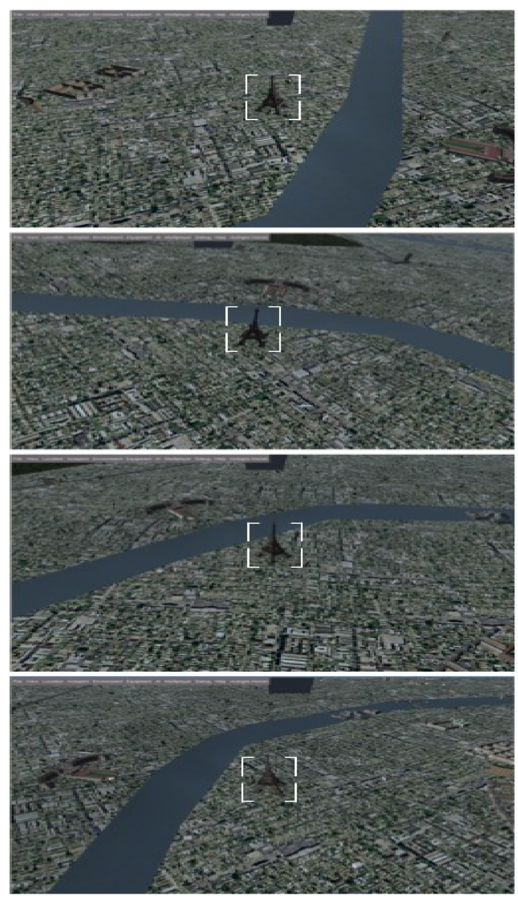


Figure 7: Keeping a close watch on the tower

Figure 8: Large view of the simulator

ACKNOWLEDGMENTS

This research was supported by the FUI AAP9 project: SHARE, and by the ANR Project GCM, program "blanche", project number NT09-504490.

REFERENCES

References

Berret, B., Darlot, C., Jean, F., Pozzo, T., Papaxanthis, C. & Gauthier, J. P. (2008). The inactivation principle: Mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements, *PLoS Computational Biology* **4**(10): 25.

Betts, J. T. (2001). *Practical Methods for Optimal Control using Nonlinear Programming*, Vol. 55, SIAM.

Boiffier, J.-L. (1998). *The dynamics of flight, the equations*, Wiley and sons.

Bullo, F. & Lewis, A. D. (2004). *Geometric Control of Mechanical Systems*, Springer-Verlag, Berlin.

Chitour, Y., Jean, F. & Mason, P. (to be published). Optimal control models of the goal-oriented human locomotion, *SIAM J. Control and Optimization* .

Chittaro, F., Jean, F. & Mason, P. (to be published). On the inverse optimal control problems of the human locomotion: stability and robustness of the minimizers, *Journal of Mathematical Sciences* .

Craighead, J., Murphy, R., Burke, J. & Goldiez, B. (2007). A survey of commercial and open source unmanned vehicle simulators, *IEEE International Conference on Robotics and Automation*.

Cummings, M. L., Platts, J. T. & Sulmistras, A. (2006). Human performance considerations in the development of interoperability standards for uav interfaces, *Moving Autonomy Forward Conference*.

Fabiani, P., Fuertes, V., Piquereau, A., Mampey, R. & Teichteil-Königsbuch, F. (2007). Autonomous flight and navigation of vtol uavs: from autonomy demonstrations to out-of-sight flights, *Aerospace Science and Technology* .

Fliess, M., Lévine, J., Martin, P. & Rouchon, P. (1995). Flatness and defect of non-linear systems: Introductory theory and examples, *International Journal of Control* **61**(6): 1327–1361.

Grewal, M. S., Weill, L. R. & Andrews, A. P. (2007). *Global Positioning Systems, Inertial Navigation, and Integration*, Wiley-Interscience.

Ippolito, C., Yeh, Y. & Campbell, C. (2009). A trajectory generation aproach for payload directed flight, 47^{th} *AIAA Aerospace Science Meeting*.

jun Yang, Z., hui Qi, X. & lin Shan, G. (2009). Simulation of flight control laws design using model predictive controllers, *Proceedings of the IEEE International Conference on Mechatronics and Automation, Changchun, China*.

Junkins, J. L. & Schaub, H. (2001). *Analytical Machanics of Aerospace Systems*, Texas A&M University.

Kim, H. J., Shim, D. H. & Sastry, S. (2002). Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles, *ACC02-AIAA1044* .

Laumond, J.-P. (1998). *Robot Motion Planning and Control*, Springer-Verlag, Berlin. Available online at http://www.laas.fr/∼jpl/book.html.

LaValle, S. M. (2006). *Planning Algorithms*, Cambridge University Press, Cambridge, U.K. Available at http://planning.cs.uiuc.edu/.

Li, W., Todorov, E. & Liu, D. (2011). Inverse optimality design for biological movement systems, *World Congress, Volume 18 Part 1*.

Moiseev, I. & Sachkov, Y. L. (2010)). Maxwell strata in sub-riemannian problem on the group of motions of a plane, *ESAIM: Control, Optimisation and Calculus of Variations* **COCV 16 (2010) 380-399**.

NATO (2008). *STANAG 4586 - B - 80 -*.

Park, S., Deyst, J. & How, J. P. (2004). *A New Nonlinear Guidance Logic for Trajectory Tracking*, number August, AIAA, pp. 1–16.

Sachkov, Y. (2011). Cut locus and optimal synthesis in the sub-riemannian problem on the group of motions of a plane, *ESAIM: Control, Optimisation and Calculus of Variations* **COCV 17 (2011) 293-321**.

Sachkov, Y. L. (2010). Conjugate and cut time in the sub-riemannian problem on the group of motions of a plane, *ESAIM: Control, Optimisation and Calculus of Variations* **COCV 16 (2010) 1018-1039**.

Schmidt, L. V. (1998). *Introduction to Aircraft Flight Dynamics*, AIAA education series.

Sheridan, T., Verplank, W. & Brooks, T. (1978). Human and computer control of undersea teleoperators, *The 14th Annual Conference on Manual Control* .

Sorton, E. F. & Hammaker, S. (2005). Simulated flight testing of an autonomous unmanned aerial vehicle using flightgear, *Infotech@aerospace, Arlington, Virginia*.

Tisdale, J., Kim, Z. & Hedric, K. (2009). Autonomous uav path planning and estimation, *Robotics and Automation Magazine, IEEE* .

Van Nieuwstadt, M. J. & Murray, R. M. (1998). Real-time trajectory generation for differentially flat systems, *International Journal of Robust and Nonlinear Control* **8**(11): 995–1020.

Wanner, J.-C. (1984). *Dynamique du vol et pilotage des avions*, Onéra - Supaéro.

wescam (2011). *www.wescam.com*.

Annexe B

Article de conférence concernant l'application des méthodes de planification sur le simulateur [4] : *"Path planning and ground control station simulator for UAV"*

L'article présenté a été accepté à la conférence *IEEE Aerospace Conference* [4].

Path Planning and Ground Control Station Simulator for UAV

Alain AJAMI, Jean-Francois BALMAT, Jean-Paul GAUTHIER*, Thibault MAILLOT
Université de Toulon, CNRS, LSIS, UMR 7296
Avenue de l'Université
83957 LA GARDE CEDEX - FRANCE
ajami@univ-tln.fr, balmat@univ-tln.fr, gauthier@univ-tln.fr, maillot@univ-tln.fr
* and INRIA TEAM "GECO"

Abstract—**In this paper we present a Universal and Interoperable Ground Control Station (UIGCS) simulator for fixed and rotary wing Unmanned Aerial Vehicles (UAVs), and all types of payloads. One of the major constraints is to operate and manage multiple legacy and future UAVs, taking into account the compliance with NATO Combined/Joint Services Operational Environment (STANAG 4586). Another purpose of the station is to assign the UAV a certain degree of autonomy, via autonomous planification/replanification strategies. The paper is organized as follows.**

In Section 2, we describe the non-linear models of the fixed and rotary wing UAVs that we use in the simulator.

In Section 3, we describe the simulator architecture, which is based upon interacting modules programmed independently. This simulator is linked with an open source flight simulator, to simulate the video flow and the moving target in 3D. To conclude this part, we tackle briefly the problem of the Matlab/Simulink software connection (used to model the UAV's dynamic) with the simulation of the virtual environment.

Section 5 deals with the control module of a flight path of the UAV. The control system is divided into four distinct hierarchical layers: flight path, navigation controller, autopilot and flight control surfaces controller.

In the Section 6, we focus on the trajectory planification/replanification question for fixed wing UAV. Indeed, one of the goals of this work is to increase the autonomy of the UAV. We propose two types of algorithms, based upon 1) the methods of the tangent and 2) an original Lyapunov-type method. These algorithms allow either to join a fixed pattern or to track a moving target.

Finally, Section 7 presents simulation results obtained on our simulator, concerning a rather complicated scenario of mission.

TABLE OF CONTENTS

1. INTRODUCTION

Recently large technological advances have been realized in the fields of UAVs in order to reduce crashes and collisions [1], [2], [3]. Human error remains the main cause, as reported by the NATO Joint Capability Group for UAVs. For this reason it is important to improve on the autonomy of UAVs via the coupling between the **carrier** (UAV) and the **payload** (camera for instance).
To meet these requirements, a strong consortium formed by several companies and research labs[1] has been created. This consortium has decided to join the R&D efforts of each partner within a project called SHARE.
SHARE project consists of developing a universal and interoperable ground control station for fixed and rotary wing UAVs and all types of payloads, with a reduced number of operators at the level of the base station.
Several technical requirements must be treated to achieve this project:

1. Increase mission effectiveness by a) controlling the trajectory of the UAV and b) merging several work stations.
2. Improve on data transmission by securing data links and reducing compression and decompression time of images.
3. Increase the universality of the ground station to support any type of UAV and payload.

Specifically LSIS lab focuses on developing a simulator of a ground control station and developing algorithms for path planning. Those algorithms must take into account payload requirements, optimal costs and obstacles (or no fight zones). The use of flight simulation tools for complex aerospace systems to reduce timetable, risk and required amount of flight testing is a well-recognized advantage of these approaches.
For these reasons we have developed a ground control station simulator which will serve several purposes:

- Simulate the behavior of the carrier, its onboard payloads and information transmitted to the control station.
- Provide support for the simulation of different missions (tracking, monitoring, etc.).
- Serve as a development platform to test several path planning algorithms.
- Serve as a demonstration platform.

The simulator is developed under the Matlab/Simulink environment. It simulates the non-linear dynamic equations

[1]Thales Alénia Space, Opéra Ergonomie, Eurocopter, ONERA, Adetel Group

978-1-4673-1813-6/13/$31.00

of UAV, payload dynamics, control module, MMI (Man Machine Interface), exterior conditions, etc.

In Section 3 we present the simulator architecture based upon interacting modules. Graphical representations used in the simulator are discussed. In 3D, to represent the video f ow and the moving target we use an open-source f ight simulator: Flightgear. Otherwise, we have developed our own 2D simulation environment to visualize UAV's trajectories. A synchronization between 2D and 3D environments is established.

In Section 5, we deal with the control module and its different layers. The control module calculates commands to send to the control surfaces of the carrier and the camera. The automation strategies of the control system allow achieving accurate and safe UAV operations. Four levels are considered for this module which offers ability to choose the control mode and the type of mission (collecting information, monitoring, tracking, etc.).

In Section 6 we present our solutions to the problem of trajectory planif cation/replanif cation for f xed wing UAV. We propose original algorithms based upon the tangent and Lyapunov vector f eld methods in order either to join a pattern or to track a moving target.
Considering a f xed target, missions consist of joining and following operator's parameterized patterns: circle, racetrack or leminscate in order to detect any activity in a specif c area and/or to collect information.

Section 7 is devoted to simulation results for several scenarios in order to test and validate our simulator and replanif cation algorithms.

In section 9, with our conclusions, we outline ideas for position estimation based upon knowledge of the target and environment information. Another idea for future is to develop a decision support system in order to aid the operator in a replanif cation context.

2. UAV MODEL

This section deals with the modeling of UAV's f ight dynamics mechanical equations. The model is the heart of the simulation process; it describes the dynamic and aerodynamic behaviors of the carrier.

For the sake of generality we try to keep generic equations. They correspond to two types of carrier: f xed and rotary wing. The simulation problems are similar in both cases however the path planning problems are much more complicated in the f xed wing case. It is the reason why we focus on this case here.
To simulate a specif c carrier it is enough to specify its aerodynamic data in a way that will be described later (Section 3).

Notations and Def nitions

$v^{(f)}$	: vector v projected in frame (f)
$X^{(f)} = (x, y, z)$	: aircraft's position vector
$(\phi, \theta, \psi)^{(f)}$	: Euler angles of aircraft (roll, pitch, yaw)
q_0, q_1, q_2, q_3	: unit attitude quaternion
$V^{(f)} = (u, v, w)^{(f)}$	: aircraft's speed vector
$V_a^{(f)} = (u_a, v_a, w_a)^{(f)}$	: aerodynamic speed vector
$\omega^{(f)} = (\omega_1, \omega_2, \omega_3)^{(f)}$	: aircraft's angular velocity vector
$F_{aero}^{(b)}$	: aerodynamic force
$F_{prop}^{(b)}$	: propulsive force
$M_{aero}^{(b)}$	: aerodynamic moment
$M_{prop}^{(b)}$	: propulsive moment
$(\delta l, \delta m, \delta n)$	: aircraft's commands (roll, pitch, yaw)
C_X, C_Y, C_Z	: aerodynamic force coeff cients
C_l, C_m, C_n	: aerodynamic moment coeff cients
m	: mass of the aircraft
$g^{(f)}$	: gravity vector (i.e. $g^{(w)} = (0, 0, g_z)$)
I	: aircraft's inertia matrix
ρ	: atmospheric density
S	: aircraft's surface of reference
l	: aircraft's length of reference
α	: aerodynamic angle of attack
β	: aerodynamic sideslip angle
$F_{hel}^{(f)}$	: force applied to the helicopter
$M_{hel}^{(f)}$	: moment applied to the helicopter
X_{mr}, Y_{mr}, Z_{mr}	: aerodynamic forces generated by main rotor
$X_{fus}, Y_{fus}, Z_{fus}$	: aerodynamic forces generated by fuselage
X_{tr}, Y_{tr}, Z_{tr}	: tail rotor forces
T	: main rotor thrust
L_{mr}, M_{mr}, N_{mr}	: aerodynamic moments generated by main rotor
L_{tr}, N_{tr}	: aerodynamic moments generated by tail rotor
v_i	: main rotor induced velocity
ω_{mr}	: main rotor rotating speed
R_{mr}	: main blade radius
$Cl_{\alpha,mr}$	: lift curve slope of the main rotor blade
b_{mr}	: main rotor blade number
c_{mr}	: main rotor blade chord length
a_s, b_s	: longitudinal and lateral tip-path-plane (TPP) f apping angle
K_{col}	: ratio of main rotor blade collective pitch to δ_{col}
K_β	: main rotor spring constant
H_{mr}	: main rotor hub's vertical position above the center of mass
P_{pr}	: prof l power of main rotor
P_i	: induced power of main rotor
P_{pa}	: parasite power of main rotor
P_c	: climbing power of main rotor
$\theta_{col,0}$	: trim offset of the main blade's collective pitch angle
$(\delta_{lat}, \delta_{lon}, \delta_{col}, \delta_{ped})$	: helicopter's commands (aileron, elevator, collective, rudder)
$(x_c, y_c, z_c)^{(f)}$	: camera's position
$(\phi_c, \theta_c, \psi_c)^{(f)}$	: angles of camera (roll, elevation, azimuth)
$(\delta\phi_c, \delta\theta_c, \delta\psi_c)$	: camera's commands

UAV Model for fixed wing

Mainly, we use the dynamical equations from [4], [5], [6]. Basically they are just the Euler equations of the solid. Modeling assumptions are the following:

1. The aircraft is a standard 6 DoF solid, symmetric w.r.t. its vertical middle plane
2. The mass and the inertia matrix of the aircraft are constant
3. The propulsion system lies along the fuselage axis
4. Atmospheric pressure is constant and effects of external temperature are neglected
5. We work in a fixed coordinate frame in which the movement and the roundness of the earth are ignored. This frame is called the "world frame".

In fact, to describe the motion we need to consider three frames:

World frame (w) This frame is attached to a reference point O. The Ox, Oy, Oz axes are oriented in the north-east-down directions

Body frame (b) This frame is attached to the aircraft's center of mass G. The $Gx^{(b)}$ axis goes along the fuselage axis. The $Gz^{(b)}$ axis is taken in the plane of symmetry of the aircraft and points downward. Thus $Oy^{(b)} = Oz^{(b)} \times Ox^{(b)}$

Aerodynamic frame (a) This frame shares its origin with frame (b). The $Gx^{(a)}$ axis is carried by the velocity vector. The rotation that transforms $Gx^{(b)}$ into $Gx^{(a)}$ depends on the angle of attack and the sideslip angle. For precise definition of the other aerodynamic coordinates see [4].

To avoid the singularities of Euler angles, we prefer to use unit quaternions in the computations. However Euler angles Roll-Pitch-Yaw parameterization is used to deliver information to the user [5].

We just show here the quaternionic rotation from frame (b) to (w):

$$R = \begin{bmatrix} 2(q_0^2+q_1^2)-1 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & 2(q_0^2+q_2^2)-1 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & 2(q_0^2+q_3^2)-1 \end{bmatrix}$$

Finally, the state vector of our system has dimension 12: position (3), velocity (3) , quaternions (3) and angular velocity (3). Due to the quaternionic representation we integrate the 13-dimensional differential system (3), (4), (5) and (6).

The six-degree-of-freedom rigid-body dynamics can be expressed by the fundamental equations, for translation and rotation:

$$m\dot{V}^{(b)} + \omega^{(b)} \times mV^{(b)} = F^{(b)}_{aero} + F^{(b)}_{prop} + mg^{(b)} \tag{1}$$

$$I\dot{\omega}^{(b)} + \omega^{(b)} \times I\omega^{(b)} = M^{(b)}_{aero} + M^{(b)}_{prop} \tag{2}$$

where

$$F^{(b)}_{aero} = \begin{bmatrix} -\frac{1}{2}\rho S V_a C_X \\ \frac{1}{2}\rho S V_a C_Y \\ -\frac{1}{2}\rho S V_a C_Z \end{bmatrix}$$

$$mg^{(b)} = \begin{bmatrix} 2(q_1q_3-q_0q_2)mg_z \\ 2(q_2q_3+q_0q_1)mg_z \\ (2(q_0^2+q_3^2)-1)mg_z \end{bmatrix}$$

$$F^{(b)}_{prop} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} F_x \\ 0 \\ 0 \end{bmatrix} \quad \text{(by applying assumption 4)}$$

$$M^{(b)}_{aero} = \begin{bmatrix} \frac{1}{2}\rho S V_a C_l \\ \frac{1}{2}\rho S V_a C_m \\ \frac{1}{2}\rho S V_a C_n \end{bmatrix}$$

$$M^{(b)}_{prop} = 0$$

$$I = \begin{bmatrix} A & -F & -E \\ -F & B & -D \\ -E & -D & C \end{bmatrix}$$

with A, B, C, D, E real constants and $F = D = 0$ by applying assumptions 1 and 2.

By applying the equation of forces (1) we deduce the acceleration vector in the frame (b)

$$\begin{aligned} \dot{u} &= -\frac{\rho}{2m} S V_a^2 C_X + \frac{F_x}{m} - \omega_2 w + \omega_3 v + 2g_z(q_1q_3 - q_0q_2) \\ \dot{v} &= \frac{\rho}{2m} S V_a^2 C_Y + \frac{F_y}{m} - \omega_1 w + \omega_3 u + 2g_z(q_2q_3 - q_0q_1) \\ \dot{w} &= -\frac{\rho}{2m} S V_a^2 C_Z + \frac{F_z}{m} - \omega_1 v + \omega_2 u + g_z(2(q_0^2+q_3^2)-1) \end{aligned} \tag{3}$$

According to the equation of moments (2) we obtain the angular acceleration vector in the frame (b):

$$\begin{aligned} (AC-E^2)\dot{\omega}_1 &= \frac{\rho l}{2} S V_a^2 (C_l C + C_n E) \\ &\quad + (BC - C^2 - E^2)\omega_2\omega_3 - E(B-A-C)\omega_1\omega_2 \\ B\dot{\omega}_2 &= \frac{\rho l}{2} S V_a^2 C_m + (C-A)\omega_1\omega_3 - (\omega_1^2 - \omega_3^2)E \\ (AC-E^2)\dot{\omega}_3 &= \frac{\rho l}{2} S V_a^2 (C_l E + C_n A) + E(B-A-C)\omega_2\omega_3 \\ &\quad + (E^2 - A(B-A))\omega_1\omega_2 \end{aligned} \tag{4}$$

The basic kinematic equation of the solid provides the motion of the quaternions:

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{5}$$

The aircraft position vector $X^{(w)}$, in the inertial frame, is given by

$$\dot{X}^{(w)} = RV^{(b)} \tag{6}$$

Aerodynamic model

The aerodynamic model consists of the data necessary to compute the external forces and moments. We have chosen to represent the aerodynamic data under the guise of lookup tables. The advantage of this choice is the representation of non-linearities. Other standard methods consist of deriving linear relation. In [7] the authors describe two methods and present the advantages of using lookup tables. It is possible to combine several approaches. In our simulator lookup tables are read and stored during the initialization step.

More precisely, we consider a set of six lookup tables which correspond to the three aerodynamic force coeff cients C_X, C_Y, C_Z and the three aerodynamic moment coeff cients C_l, C_m, C_n. These coeff cients depend on angle of attack, sideslip angle and f ap def ection. Equations of force coeff - cients are:

$$
\begin{aligned}
C_X &= C_{x_0} + k_i C_Z^2 \\
C_Y &= C_{y_\beta}\beta + C_{y_p}\frac{\omega_1 l}{V_a} + C_{y_r}\frac{\omega_3 l}{V_a} + C_{y_{\delta n}}\delta n \\
C_Z &= C_{z_\alpha}\alpha + C_{z_q}\frac{\omega_2 l}{V_a} + C_{z_{\delta m}}\delta m
\end{aligned}
$$

Equations of moment coeff cients are:

$$
\begin{aligned}
C_l &= C_{l_\beta}\beta + C_{l_p}\frac{\omega_1 l}{V_a} + C_{l_r}\frac{\omega_3 l}{V_a} + C_{l_{\delta n}}\delta n + C_{l_{\delta l}}\delta l \\
C_m &= C_{m_\alpha}\alpha + C_{m_q}\frac{\omega_2 l}{V_a} + C_{m_{\delta m}}\delta m \\
C_n &= C_{n_\beta}\beta + C_{n_p}\frac{\omega_1 l}{V_a} + C_{n_r}\frac{\omega_3 l}{V_a} + C_{n_{\delta n}}\delta n + C_{n_{\delta l}}\delta l
\end{aligned}
$$

Coeff cients $C_{x_0}, C_{y_\beta}, C_{z_\alpha}, C_{l_\beta}, C_{m_\alpha}, C_{n_\beta}$ are also tabulated with respect to values of the f ap def ections.
Standard interpolation methods are used to extract the data from these tables.

UAV Model for rotary wing (sketch)

Our model for a rotorcraft (single main rotor with tail rotor) is detailed in [8], [9].
Again from the fundamental equations (1) and (2) we obtain a similar 12-dimensional differential system. Assumptions made in section 2 still hold true.

The equation of forces and moments which are applied to the carrier are respectively given by:

$$
F_b = \begin{bmatrix} F_{bx} \\ F_{by} \\ F_{bz} \end{bmatrix} = \begin{bmatrix} X_{mr} + X_{fus} \\ Y_{mr} + Y_{fus} + Y_{tr} \\ Z_{mr} + Z_{fus} \end{bmatrix}
$$
$$
M_b = \begin{bmatrix} M_{bx} \\ M_{by} \\ M_{bz} \end{bmatrix} = \begin{bmatrix} L_{mr} + L_{tr} \\ M_{mr} \\ N_{mr} + N_{tr} \end{bmatrix}
$$

To compute the main rotor forces and moments, we f rst calculate the main rotor thrust T and the induced velocity v_i:

$$
T = \frac{\rho\omega_{mr}R_{mr}^2 Cl_{\alpha,mr} b_{mr} c_{mr}}{4} \cdot \left(w_a + a_s u_a - b_s v_a + \frac{2}{3}\omega_{mr}R_{mr}(K_{col}\delta_{col} + \theta_{col,0}) - v_i \right)
$$

$$
v_i^2 = \sqrt{\left(\frac{\hat{v}^2}{2}\right)^2 + \left(\frac{T}{2\rho\pi R_{mr}^2}\right)^2} - \frac{\hat{v}^2}{2}
$$

where, $\hat{v}^2 = u_a^2 + v_a^2 + (w_a + a_s u_a - b_s v_a)(w_a + a_s u_a - b_s v_a - 2v_i)$.

Next, we compute the forces and moments generated by the main rotor as follows:

$$
\begin{aligned}
X_{mr} &= -T\sin(a_s) \\
Y_{mr} &= T\sin(b_s) \\
Z_{mr} &= -T\cos(a_s)\cos(b_s) \\
L_{mr} &= (K_\beta + TH_{mr})\sin(b_s) \\
M_{mr} &= (K_\beta + TH_{mr})\sin(a_s) \\
N_{mr} &= -(P_{pr} + P_i + P_{pa} + P_c)/\omega_{mr}
\end{aligned}
$$

The f nal differential system is given by (3), (4), (5), (6) with the previous values of forces and moments.

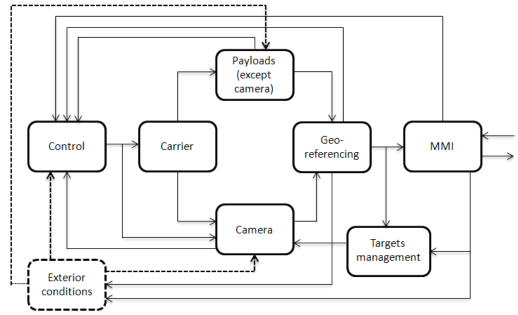


Figure 1. Module based architecture diagram

3. Simulator Architecture

The goal is to def ne a completely modular architecture allowing operators to easily add, remove or modify some parts of the simulator while maintaining the real characteristics of a ground control station. The user can benef t from the help of assistants modules to add or modify the aerodynamic coeff cients and "payload" models.

This system must be able to simulate all the operational environment of the station, in order to test and verify the development of this one. It should allow to validate our autonomous path planning algorithms. Furthermore, the simulator can be used to train operators or as a portable demonstrator. The architecture must take into account all these aspects.

In this subsection, we brief y describe the different modules of the simulator. A schematic of the interactions between each module is shown in Fig.1.

The module *carrier* ensures the resolution of the UAV's equations of motion described in section 2. The outputs of the controller and the exterior conditions are the inputs of this module. The output consists of the state vector, as a continuous function of time.

The module *Camera* deals with the camera's equations of motion. The model tracks the orientation of the camera's line of sight. The camera is assumed to be rigidly attached to the center of mass of the aircraft.
For the camera, we use the following simple first order model:

$$x_c = x^{(w)}$$
$$y_c = y^{(w)}$$
$$z_c = z^{(w)}$$
$$\dot{\phi}_c^{(w)} = \omega_1 - \frac{1}{\tau}\left(\phi_c^{(w)} - \phi^{(w)}\right) + \frac{1}{\tau}\delta\phi_c$$
$$\dot{\theta}_c^{(w)} = \omega_2 - \frac{1}{\tau}\left(\theta_c^{(w)} - \theta^{(w)}\right) + \frac{1}{\tau}\delta\theta_c$$
$$\dot{\psi}_c^{(w)} = \omega_3 - \frac{1}{\tau}\left(\psi_c^{(w)} - \psi^{(w)}\right) + \frac{1}{\tau}\delta\psi_c$$

The variable τ defines the time constant of the camera steering system. The input of this module is the state vector of the UAV, the controls, the target data and exterior conditions. The output is the state vector of camera.

The *Control* module ensures the control of the UAV and the camera. It has several modes and levels of control. We detail this module in section 5. The input of this module is the sate vector of the UAV and the camera, the exterior conditions, the geo-referencing data and the set points given by the user via the MMI. The output provides the controls of carrier and camera.

The *Exterior conditions* module manages weather conditions, more particularly wind. We use a Von Karman model [10] to generate the wind velocity vector required by the resolution of the equations of motion. Indeed, the wind speed V_w allows us to calculate aerodynamic speed vector V_a and deduce thereafter aerodynamic angles α and β from geometric relationships between the speed vectors $V_w^{(b)}$ and $V^{(b)}$.

MMI (Man Machine Interface) manages the flow of information between the operator and the UAV. This unit allows the user to control the carrier and the payload, and to define the mission to achieve (monitoring, collecting information, tracking, etc.). Also, it displays the flow of the video and positions of the UAV and the targets.

Geo-referencing provides the synchronization between 3 display modes: the 2D map, the 3D environment and the radar representation (graphical representation is discussed in the next subsection). It synchronizes in real-time the trajectory of the aircraft, the relative positions of possible targets and the actual video flow on different types of maps.

Payloads unit includes all payloads except camera. It includes simulation of noise and bias on payloads.

The Targets management unit allows the user to manage the fixed or mobile targets (coordinates, speed, trajectory, etc.).

Graphical representation

3D representation—A brief review of the significant features of 14 simulators (Flightgear, X-Plane, Simbad, Matlab, etc.) is presented in paper [11]. In order to evaluate the simulators

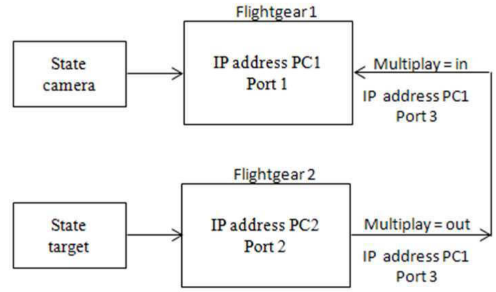


Figure 2. Flightgear configuration

from the perspective of a robotics researcher, the authors propose four criteria:

- Physical Fidelity
- Functional Fidelity
- Ease of Development
- Cost

On the basis of this evaluation, we decided to work with Matlab/ Simulink and Flightgear. Matlab/Simulink is a numerical simulation environment which can communicate with other simulation environments that may provide better physics simulation and visualization. Also, it offers opportunity to work with the S-function language that allows programming separately the different modules and it is able to produce ordinary C programs.

Flightgear has been chosen to simulate in real-time the virtual environment and the video flow. It is open-source so that the code can be adapted as needed. It runs on many different operating systems (Microsoft Windows and Linux) and it interfaces nicely with Matlab/Simulink.

To simulate multiple objects (camera, targets, etc.) in the same Flightgear's instance, servers or other computers are required. Hence it is possible to run multiple instances of Flightgear, the counterpart being the management of computer communications. In our case we have used two synchronized computers and we have added the Multiplay's function to the script of Flightgear. As shown in Fig.2, we have defined the receiving PC (Multiplay = in) and sending PC (Multiplay = out) with corresponding port number.

2D representation—The UAV trajectory is displayed on a 2D map (see Fig.3). Of course, connection and synchronization of the map with the Flightgear environment must be ensured. For this purpose, we could have used the open-source software Atlas. Actually the choice between open-source or commercial software can be discussed. Atlas viewing is open-source application which can be connected directly to Flightgear. This application is a simple and free 2D map; it allows displaying aircraft's current location on a moving map [12], [13].
However, Atlas is limited compared to our needs: 1. dynamic interaction with the map (definition/redefinition of waypoints by clicking directly on the map); 2. display orientation of the camera's line of sight; etc.

To meet these requirements, we have chosen to develop our own 2D map application on Matlab/Simulink. For this we have designed a map by using aviation charts [14] to display the aircraft trajectory and waypoints.

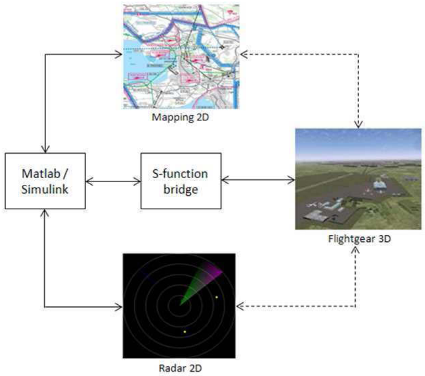


Figure 3. 2D and 3D graphical representation

On the other hand, we wanted to incorporate a radar view in our simulator, providing information about the relative positions of possible targets. A schematic showing relationships between the set of display screens is depicted in Fig.3.
The S-function “Bridge” ensures the communication between Matlab/Simulink and Flightgear.

4. Man machine interface

The set of simulator’s features are managed thanks to the Man Machine Interface (MMI) as shown in Fig.4. The user can therefore control and assess the state of the system. This interface deals with the information f ow between the operator and UAV. The graphical interface proposes several windows which allow to set the parameters of simulation and to visualize the state of the system in real time.

The MMI allows also initializing the simulator. This initialization phase is required before any operation. It permits to def ne parameters that are specif c to a given simulation instance. The user selects a UAV type (f xed or rotary wing) among a data base. The aerodynamic model, the corresponding aerodynamic parameters and the model of payload are loaded accordingly.

Finally the set of maps corresponding to the area where the scenario takes place, and the initial state of the aircraft are selected. Fig.4 shows a part of our MMI. The user can conf gure online simulation according to a predef ned scenario according to the following parameters:

- Waypoints table for the f ight plan. The table can be completed in two distinct ways: entering the coordinates manually or directly clicking on the 2D map to automatically retrieve the coordinates of the points
- Navigation mode (navigation by waypoints, tracking, pattern of monitoring, etc.)
- UAV control mode and camera control mode (manual or automatic)
- Geometrical patterns (circle radius, length and orientation of the racetrack, etc.)
- Target management (position in case of a f xed target, speed and direction when a moving target)
- Weather conditions: speed and direction of the wind.

At this point, let us recall that one of our main purposes is to give the UAV a certain degree of autonomy (replanif cation). For instance when the initial mission is interrupted and when there is the need to determine a new path online, the user can invoke certain replanif cation algorithms. Our algorithms are mostly based on Lyapunov-type method or on the Pontryagin Maximum Principle (PMP) [15]. We have chosen to minimize either the time or a length-curvature cost. Manual replanif cation can also be achieved by simply clicking on the map to create new waypoints. The choice between the different solutions is performed via MMI.

At the level of the user the MMI displays several simulation pieces of information, such as:

- Flight data (position, heading, speed, etc.)
- 3D video Flow
- UAV 2D trajectory
- Radar display
- Waypoints list

5. Automatic control

In this section, a description of the various control modules is proposed. The automation level is more or less signif cant depending on the involvement degree of the operator in the decision process. The control action can be performed by humans or computers. Thus, according to [16], [17], various levels of automation can be incorporated in a decision making system.

The purpose of the control system at the lowest level is to generate control signals based upon position and attitude of the aircraft in order to satisfy the requirements of longitudinal and lateral maneuverability.

The control module dedicated to the f xed wing model is depicted in Fig.5. This module is divided into four distinct hierarchical layers: f ight path, navigation controller, autopilot and f ight control surfaces controller. The top level of the control architecture manages the f ight path. It encapsulates all the algorithms that generate a set of waypoints $P_n = (\xi_n, V_n, T_n)_{n\geq 0}$ with

$$\begin{aligned} \xi_n &\in \mathbb{R}^3 : \text{world coordinates of the point to be reached} \\ V_n &\in \mathbb{R}^+ : \text{desired speed between } P_n \text{ and } P_{n-1} \\ T_n &\in \mathbb{R}^+ : \text{latency above the point (in the case of a helicopter)} \end{aligned}$$

The navigation controller transforms the waypoints P_n into a desired velocity command V_d, a desired altitude command h_d and a desired heading command ψ_d.

The autopilot receives these commands and calculates heading, airspeed and altitude holders. The longitudinal and lateral dynamics of the carrier are considered decoupled so that the longitudinal and lateral autopilots are independent [6].

Using the measured altitude, speed w and angular velocity ω_2, the longitudinal autopilot gives the elevator def ection.
The lateral autopilot depends on the desired heading, pitch angle ϕ, and angular velocity ω_1. The output is the heading command.
A classical PID controller is enough for each autopilot.

Finally, level zero consists of a direct access to the control surfaces of the UAV. Although this task is unrealistic we kept this possibility for internal tests (or for the fun).
User can choose among three different modes for both the UAV and the camera turret:

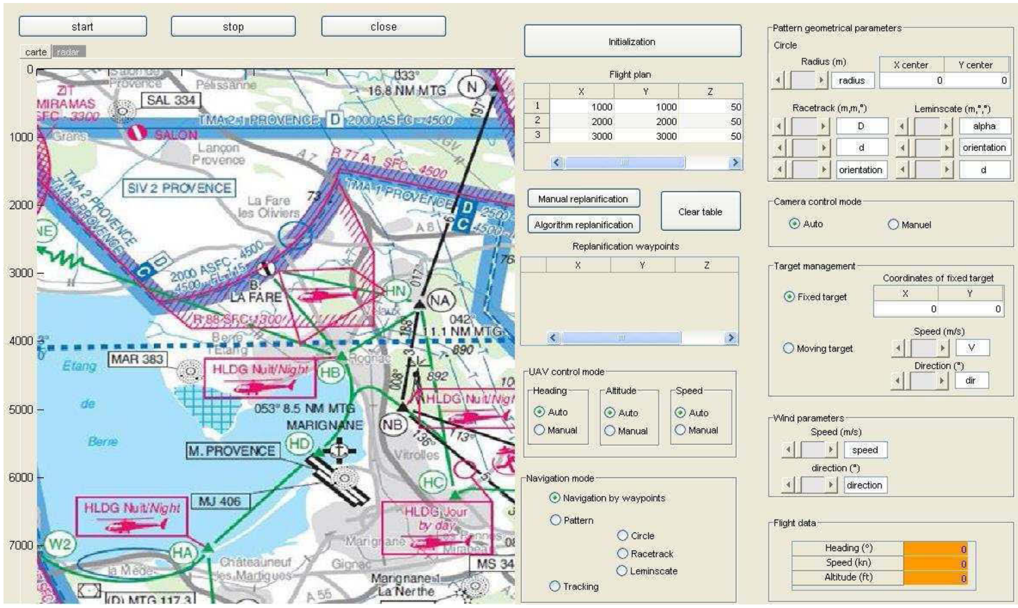


Figure 4. The Man Machine Interface of the simulator

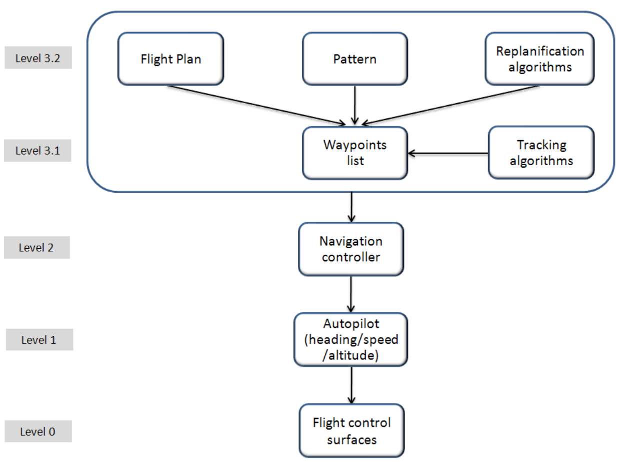


Figure 5. Level based decomposition of the controller

Manual mode. A joystick allows to directly control the heading, speed and altitude commands. The azimuth and elevation commands of the camera can also be controlled.
Semi-manual mode. In this case, at least one of the three components of the heading, speed and altitude command is managed by the system.
Automatic mode. The UAV is fully autonomous and the system manages automatically the missions specif ed by the user.

6. Path planning

In this section, we present brief y our methodology for the trajectory planif cation/replanif cation for f xed wing UAV (Again, the rotary wing case is more classical and in fact easier, and we have developed a general methodology in a series of papers [18], [19], [20], [21], [22], [23]).
In f xed wing case, we present two path planning approaches that can be used in replanif cation in order to join a pattern or to track targets. The results will be stated without proof since they will be the subject of a forthcoming paper. The tangent vector f eld guidance method and the Lyapunov method are used for UAV's f ight guidance.
These approaches are limited by different constraints such as the geographical conf guration, the UAV's dynamic characteristics, the UAV's initial orientation and the traveling direction of the pattern.
Finally, we propose to use an extended Kalman f lter for target motion prediction.

Tangent vector f eld guidance

We present a dynamic path planning algorithm for a UAV based on the tangent vector f eld guidance law that allows monitoring and tracking a ground target. This algorithm calculates a path by taking into account the physical constraints, the initial position of UAV and the position of target [24].

In Fig.6, UAV and target are represented in a topographical view. The UAV direction vector is denoted $V^{(w)}$. The goal is to join tangentially a circle with radius r around the target from any initial direction of the UAV.

From the position of UAV we calculate two tangents to the target circle $T1$ and $T2$. ψ_1, ψ_2 are the angles between the direction vector $V^{(w)}$ and $T1$, $T2$ respectively. In the case where the traveling direction along the f nal pattern is not def ned, the UAV should head the nearest tangent. An example is depicted in Fig.7.
The new UAV direction angle ψ_d can be determined as:

$$\psi_d = \min(\psi_1, \psi_2)$$

The particular case when $\|D\| < r$ is treated as follows: we simply allow the UAV to go straight ahead maintaining its initial direction. When the UAV is outside the circle, we apply again the algorithm. Notice that when the UAV starts far from the target, this strategy coincides exactly with the minimum time optimal policy [25].

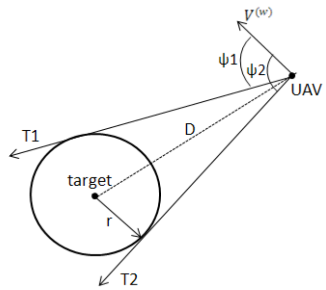


Figure 6. From the position of UAV we calculate two tangents on the circle of radius r around the target

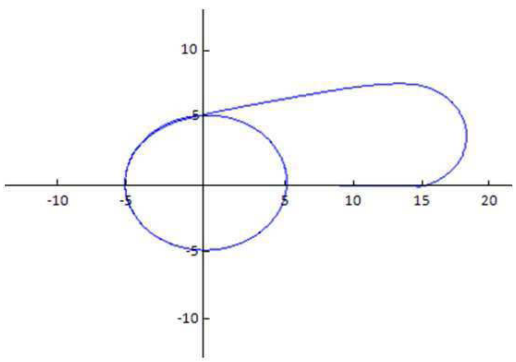


Figure 7. Trajectory resulting from the tangent vector f eld guidance starting at the point (15;0) with a direction of 0 radian and arriving tangent to a pattern, here the circle centering at (0,0)

Lyapunov vector f eld guidance

In this section we consider the problem from the kinematic point of view only. In particular we consider that the aircraft commands are identically equal to angular velocities, i.e $(\omega_1, \omega_2, \omega_3) \equiv (\delta l, \delta m, \delta n)$. We deal successively with the 2D and 3D problem. The 2D problem is for instance the case of a HALE UAV at constant altitude.

The Lyapunov method has been used by several authors, in several different ways [24], [26]. Here we present an original Lyapunov strategy, serving the advantage to be completely smooth, and very simple to apply. Frequently Lyapunov strategies presented in the litterature are discontinuous.

First 2D problem—Our goal is to stabilize the UAV on a predef ned pattern (a circle) in the f ying plan above the target. From the kinematic point of view, the equations of the motion are just the standard Dubins equations (see [27] for a justif - cation), i.e.

$$\begin{cases} \dot{x} = \cos\psi \\ \dot{y} = \sin\psi \\ \dot{\psi} = \omega_3, \end{cases} \tag{7}$$

with $q = (x, y, \psi) \in \mathbb{R}^2 \times \mathbb{S}^1$ being the state (where $(x, y) \in \mathbb{R}^2$ is the UAV's coordinates vector in the constant altitude plane, ψ being the yaw angle), and ω_3 being the control variable. The yaw control ω_3 is constrained by $\omega_{3\,\min} \leq \omega_3 \leq \omega_{3\,\max}$, with $\omega_{3\,\max} > 0$ and $\omega_{3\,\max} > \omega_{3\,\min}$.

And we f x the target pattern to be a circle $\mathcal{C}$:

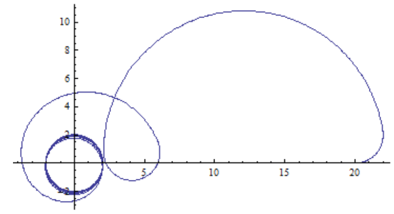


Figure 8. LaSalle's principle based planif cation results in 2D

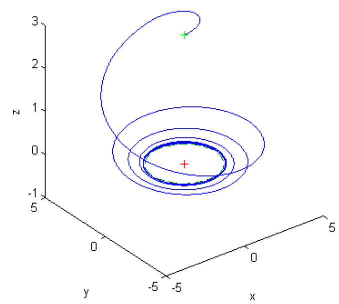


Figure 9. LaSalle's principle based planif cation results in 3D

- centered at the origin with radius $r = 1/\omega_{3\,\max}$;
- traveled counterclockwise.

Let us observe that $\mathcal{C}$ corresponds to the maximal curvature that can be achieved by the system.

We have the following result. Proof will be develop in a forthcoming paper.

Theorem 6.1: Therexists an explicit feedback control function $u(\cdot)$ which has the following properties:

- $u(\cdot)$ is $\mathcal{C}^\infty$;

- The pattern $\mathcal{C}$ is a globaly asymptotically stable attractor for the closed-loop system that result from applying $u(\cdot)$.

Remark 6.2: In other Lyapunov type methods there are in general the 2 following drawbacks:

1. The synthesis is not smooth

2. In our method the pattern which is asymptotically reached is exactly the pattern $\mathcal{C}$. In general it is not the case. For other Lyapunov technique f nal curvature is a bit smaller than the maximum curvature.

Secondly 3D problem—Now we consider a MALE UAV f ying at constant speed. Our goal does not change: we want to stabilize the UAV on a predef ned pattern above the target.

Equations of motion are def ned on section 2. We considered that the UAV is controlled by three commands on angular velocity acting physically on three angles: roll, pitch and yaw def ned by:

- ω_1, the roll control constrained by $\omega_{1\,\min} \leq \omega_1 \leq \omega_{1\,\max}$,

with $\omega_{1\,\max} > 0$ and $\omega_{1\,\max} > \omega_{1\,\min}$;
• ω_2, the pitch control constrained by $\omega_{2\,\min} \le \omega_2 \le \omega_{2\,\max}$, with $\omega_{2\,\max} > 0$ and $\omega_{2\,\max} > \omega_{2\,\min}$;
• ω_3, the yaw control constrained by $\omega_{3\,\min} \le \omega_3 \le \omega_{3\,\max}$, with $\omega_{3\,\max} > 0$ and $\omega_{3\,\max} > \omega_{3\,\min}$.

Here, we fix the pattern as the circle $\mathcal{C}_{3D}$:

• centered at the origin with radius $r = 1/\omega_{3\,\max}$ and contained in a horizontal plane;
• traveled counterclockwise.

As for the 2D case we have the following result.

Theorem 6.3: There exists three explicit control functions $u^i_{3D}(\cdot)$, $i \in \{1, 2, 3\}$, for each UAV's controls, which have the following properties:

• $u^i_{3D}(\cdot)$ is $\mathcal{C}^\infty$, $i \in \{1, 2, 3\}$;

• The pattern $\mathcal{C}_{3D}$ is a globaly asymptotically stable attractor for the closed-loop system that result from applying the three control functions.

Remark 6.4: Again one strong point is that the attractor $\mathcal{C}_{3D}$ is exactly reached.

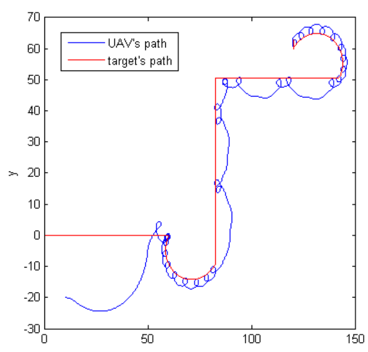


Figure 10. LaSalle's principle based tracking results in 2D when the target velocity is close to the UAV's velocity

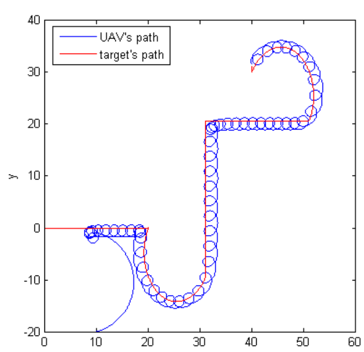


Figure 11. LaSalle's principle based tracking results in 2D when the target velocity is much smaller than UAV's velocity

Numerical results— We show numerical simulation using control functions of both Theorem 6.1 and 6.3.

First we present an example of a Dubin's system simulation using the feedback control law of Theorem 6.1. Fig.8 shows a trajectory starting at the point (20,0) with a direction of 0 radian and reaching circle $\mathcal{C}$.

Numerical result of Theorem 6.3 is showned on Fig.9. Here the UAV starts at the point (0,0,3) with an orientation of 0 radian for the pitch, roll and yaw. The trajectory reach the circle $\mathcal{C}_{3D}$.

Tracking of a moving target—Algorithm shown in previous section gives some results when the target is fixed in location. Based on this algorithm we have developed another one algorithm to track a moving target. The idea is to readjust the pattern's position according to the target's position which can be known or estimated.
To estimate the target position (when the target trajectory is unknown) we use an extended Kalman filter. It predicts the target velocity vector and estimates its future position.

The simple case of a HALE UAV tracking a target with a priori known movement is shown on Fig.10 and 11 . The UAV starts horizontaly at the position $(10, -20, 0)$ and the target at the origin. We present two cases depending on the difference between the target's velocity and the aircraft's velocity. Fig.10 shows result when target's velocity is close to the aircraft's velocity and Fig.11 shows a trajectory of a target having a lower speed.
Fig.12 shows result of tracking a target following an hippodrome pattern. The UAV starts horizontaly at the position $(20; 30; 10)$. In this example we use an extended Kalman filter to estimate the future position of the target.

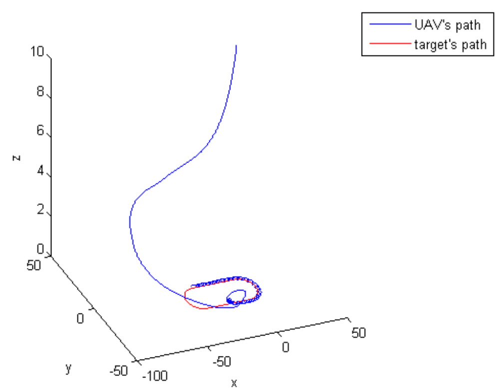


Figure 12. LaSalle's principle based tracking results in 3D

7. Simulation results

In this section we present simulation results of a flight scenario for a fixed wing UAV on our simulator. The assumptions are:

• the UAV has a sufficient reserve of fuel
• the UAV moves at a fixed speed and constant altitude
• the speed of the target is assumed to be fixed and less than the minimum speed of the UAV
• wind speed is fixed at 5 m/s throughout the simulations.

The Fig.13 depicts a flight scenario which implements several phases of the path planning and replanification.

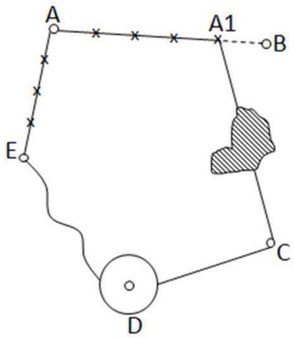


Figure 13. Diagram of complete scenario

The scenario is as follows:

$A \rightarrow B$: (A) is the starting point. The UAV travels through the waypoints of a f ight plan previously prepared.
$A1 \rightarrow C$: Before arriving at destination (B), the operator decides to divert the UAV and determines a new point of interest (C). In practice, the user clicks on the map or specif es manually coordinates of (C). The operator selects the replanif cation algorithm to accomplish this task, knowing that there is a no-f y-zone on the way.
$C \rightarrow D$: The operator decides to go to watch a stationary target located at (D). He chooses to turn around it by realizing a circle pattern.
$D \rightarrow E$: At (D) a moving target is identif ed, moving in the direction of (E), the operator decides to track it.
$E \rightarrow A$: The operator decides to end the mission and to return the UAV to its home base.

Below, we present the simulation results by detailing each step.

$A \rightarrow B$: See Fig.14. The UAV starts its f ight with an initial heading of 0 radians and following 3 waypoints spaced 2 km at a speed of 80 knots. It passes through these points straightly.

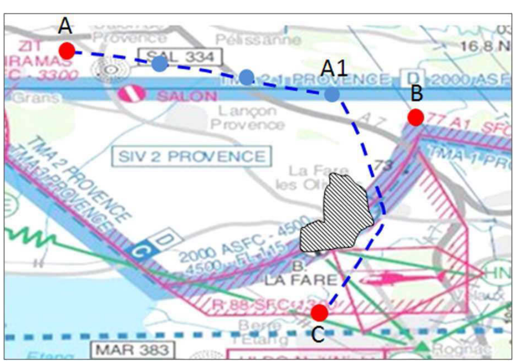


Figure 14. UAV trajectory between point A and C. UAV trajectory resulting of navigation by waypoint ($A \rightarrow A1$) and Pontryagin Maximum Principle ($A1 \rightarrow C$)

$A1 \rightarrow C$: (Fig.14 again) At (A1) the UAV changes direction and moves to (C). To f nd an optimal trajectory and to avoid the obstacle on the way, we use a path planning algorithm based on the Pontryagin Maximum Principle (PMP). The PMP allows computing an optimal trajectory with a minimization of a cost function which is in our case the trade-off between curve length and curvature.

$C \rightarrow D$: In Fig.15 the purpose is to go monitoring a f xed target located at (D). For this we use the algorithm of tangent vector f eld guidance given in section 6 allowing the UAV to reach a pattern tangentially and we start describing a circle centered at (D) with a radius of 1km.

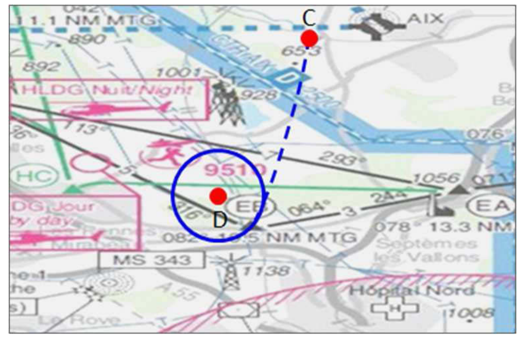


Figure 15. UAV trajectory resulting of the tangent vector f eld guidance starting at point C and arriving tangentially at the pattern centered at point D.

Figure 16. Camera view of the f xed target.

$D \rightarrow E$: Fig.17 shows the trajectory of the UAV tracking a moving target which moves straight at a constant speed 45 knots and less the minimum speed of the UAV which moves at 70 knots (its minimum speed). To achieve this mission we use the tracking algorithm based on the Lyapunov vector f eld guidance algorithm presented in Section 6.

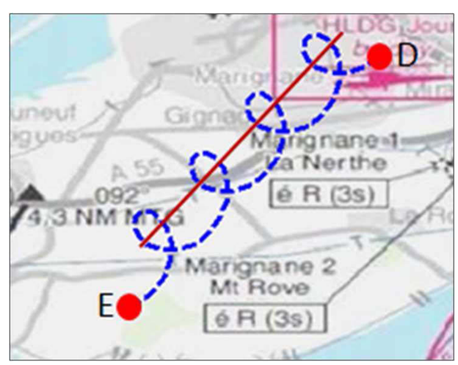


Figure 17. UAV trajectory resulting of the Lyapunov vector f eld guidance starting at point D and tracking a target moving in a straight line (red line) at a constant speed.

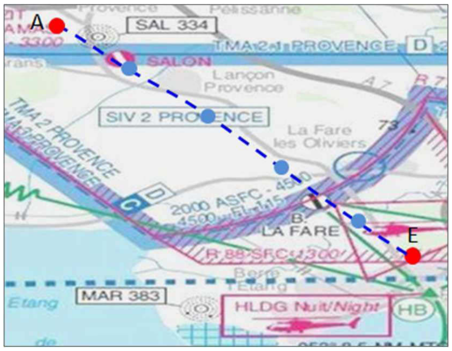


Figure 18. Path back to the base following the waypoints defined online

$E \to A$: In Fig.18, the UAV returns back in navigation mode and follows waypoints set to return to the base.

These simulations allowed us to test and validate our simulator (model, control module, MMI, synchronization between different mode of view, etc.) through a reasonable scenario and to test our replanification algorithms.

8. Conclusion

In this paper we presented a ground control station simulator for fixed and rotary wing. The simulator's architecture is based upon modules programmed independently, allowing modularity and easy maneuverability. As a consequence the simulator is able to simulate any UAV model.

Besides the well known advantages of the modularity, the operator can select simulation parameters and visualize the behavior of UAV and target in a virtual environment, thanks to MMI interface.
The simulator allowed us to simulate and validate all the operational environment of the station. Furthermore, the simulator could be used as a portable demonstrator, or for the purpose of operators training.

Another part of this paper concerns the path planification/replanification algorithms for fixed wing UAVs. Actually, our simulated station is assumed to provide a certain degree of autonomy to the UAV.
We developed certain original path planning algorithms in 2D and 3D.

Finally, we presented simulation results for scenarios of missions. To conclude this work we display in Fig.19 a view of the ground control station simulator.

9. Future works

One of the goals of this work is to develop a ground control station simulator and to define efficient algorithms for trajectory planification/replanification. In previous subsections, we have presented two path planning algorithms and an Extended Kalman Filter has been chosen for the target motion estimation.

One of the future work directions concerns the improvement of estimated location of the target. In this context, the idea is to improve on target tracking with road network information. Many studies [24], [28], [29] deal with this problem and different approaches are proposed. For the target motion prediction the authors use particle filters and a motion target model can be used depending on the target characteristics.

Another idea is to develop a decision support system in order to aid the operator in a replanification context of mission. This system should allow the operator to take the best decision under a set of rules/constraints. To propose a new path, this system should take into account safety information like fuel rate, weather conditions, risk zones, time objectives and environment parameters. Several approaches are possible for this purpose, such as fuzzy logic, bayesian networks, neural networks or multicriteria methods [30], [31], [32], [33].

Acknowledgments

This research was supported by the FUI AAP9 project: SHARE, and by the ANR Project GCM, program “blanche”, project number NT09-504490.

References

[1] J. Tisdale, Z. Kim, and J. Hedrick, “Autonomous uav path planning and estimation,” *Robotics Automation Magazine, IEEE*, vol. 16, no. 2, pp. 35 –42, june 2009.

[2] C. Ippolito, Y. Yeh, and C. Campbell, “A trajectory generation approach for payload directed flight,” *47th AIAA Aerospace Science Meeting*, 2009.

[3] P. Fabiani, V. Fuertes, A. Piquereau, R. Mampey, and F. Teichteil-Knigsbuch, “Autonomous flight and navigation of vtol uavs: from autonomy demonstrations to out-of-sight flights,” *Aerospace Science and Technology*, vol. 11, no. 23, pp. 183 – 193, 2007.

[4] J.-L. Boiffier, *The dynamics of flight. The equations*. John Wiley and sons, 1998.

[5] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*. AIAA (American Institute of Aeronautics & Ast, 2003.

[6] R. Nelson, *Flight Stability and Automatic Control*. McGraw-Hill, 1998.

[7] R. W. DETERS, G. A. DIMOCK, and S. M. S., “Icing encounter flight simulator,” *American Institute of Aeronautics and Astronautics*, vol. 43, pp. 1528–1537, 2006.

[8] A. Bramwell, G. Done, and D. Balmford, *Bramwell’s Helicopter Dynamics, Second Edition*. AIAA (American Institute of Aeronautics & Ast, 2001.

[9] G. Cai, B. Chen, and T. Lee, *Unmanned Rotorcraft Systems*. Springer, 2011.

[10] W. Leithead, S. de la Salle, and D. Reardon, “Role and objectives of control for wind turbines,” *Generation, Transmission and Distribution, IEE Proceedings C*, vol. 138, no. 2, pp. 135 –148, mar 1991.

[11] J. Craighead, R. Murphy, J. Burke, and B. Goldiez, “A survey of commercial open source unmanned vehicle simulators,” in *Robotics and Automation, 2007 IEEE International Conference on*, april 2007, pp. 852 –857.

[12] (2004, September) Atlas home page. [Online]. Available: http://atlas.sourceforge.net/

[13] A. Shaw and D. Barnes, “Landmark recognition for localisation and navigation of aerial vehicles,” in *Intelligent Robots and Systems, 2003. (IROS 2003). Pro-*

Figure 19. Large view of the simulator

ceedings. 2003 IEEE/RSJ International Conference on, vol. 1, oct. 2003, pp. 42 – 47 vol.1.

[14] [Online]. Available: http://www.sia.aviation-civile.gouv.fr/

[15] R. Vinter, *Optimal control*, ser. Systems & Control: Foundations & Applications. Boston, MA: Birkhäuser Boston Inc., 2000.

[16] M. L. Cummings, J. T. Platts, and A. Sulmistras, "Human performance considerations in the development of interoperability standards for uav interfaces," in *Moving Autonomy Forward Conference*, 2006.

[17] T. B. Sheridan and W. L. Verplank, "Human and computer control of undersea teleoperators," in *The 14th Annual Conference on Manual Control*, 1978.

[18] J.-P. Gauthier and V. Zakalyukin, "On the codimension one motion planning problem," *Journal of Dynamical and Control Systems*, vol. 11, no. 1, pp. 73–89, Jan. 2005. [Online]. Available: http://dx.doi.org/10.1007/s10883-005-0002-6

[19] ——, "On the motion planning problem, complexity, entropy, and nonholonomic interpolation," *Journal of Dynamical and Control Systems*, vol. 12, no. 3, pp. 371–404, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1007/s10450-006-0005-y

[20] ——, "Entropy estimations for motion planning problems in robotics," *Volume In honor of Dmitry Victorovich Anosov, Proceedings of the Steklov Mathematical Institute*, vol. 256, pp. 62–79, 2007.

[21] J. Gauthier, B. Berret, and V. Zakalyukin, "Nonholonomic interpolation for kinematic problems, entropy and complexity," *Mathematical Control Theory and Finance*, pp. 187–210, 2008.

[22] J. Gauthier, B. Jakubczyk, and V. Zakalyukin, "Motion planning and fastly oscillating controls," *SIAM Journal on Control and Optimization*, vol. 48, no. 5, pp. 3433–3448, 2010. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/090761884

[23] N. Boizot and J.-P. Gauthier, "Motion planning for kinematic systems," 2012.

[24] H. Chen, K. Chang, and C. Agate, "Tracking with uav using tangent-plus-lyapunov vector f eld guidance," in *Information Fusion, 2009. FUSION '09. 12th International Conference on*, july 2009, pp. 363 –372.

[25] A. A. Agrachev and Y. L. Sachkov, *Control theory from the geometric viewpoint*, ser. Encyclopaedia of Mathematical Sciences. Berlin: Springer-Verlag, 2004, vol. 87, control Theory and Optimization, II.

[26] M.-D. Hua, T. Hamel, P. Morin, and C. Samson, "A control approach for thrust-propelled underactuated vehicles and its application to vtol drones," *Automatic Control, IEEE Transactions on*, vol. 54, no. 8, pp. 1837 –1853, aug. 2009.

[27] H. Chitsaz and S. LaValle, "Time-optimal paths for a dubins airplane," in *Decision and Control, 2007 46th IEEE Conference on*, dec. 2007, pp. 2379 –2384.

[28] K. Benameur, B. Pannetier, and V. Nimier, "A comparative study on the use of road network information in gmti tracking," in *Information Fusion, 2005 8th International Conference on*, vol. 1, july 2005, p. 8 pp.

[29] Y. Chen, V. Jilkov, and X. Li, "On-road target tracking using radar and image sensor based measurements," in *Information Fusion, 2009. FUSION '09. 12th International Conference on*, july 2009, pp. 2091 –2098.

[30] J.-F. Balmat, F. Lafont, R. Maifret, and N. Pessel, "A decision-making system to maritime risk assessment," *Ocean Engineering*, vol. 38, no. 1, pp. 171 – 176, 2011.

[31] M. Cummings, J. Marquez, and N. Roy, "Human-automated path planning optimization and decision support," *International Journal of Human-Computer Studies*, vol. 70, no. 2, pp. 116 – 128, 2012.

[32] J. F. Smith III and T. H. Nguyen, "Fuzzy decision trees for planning and autonomous control of a coordinated team of uavs," pp. 656 708–656 708–12, 2007. [Online]. Available: + http://dx.doi.org/10.1117/12.716896

[33] P. Wu, P. Narayan, D. Campbell, M. Lees, and R. Walker, "A high performance fuzzy logic architecture for uav decision making." in *Computational Intelligence*, B. Kovalerchuk, Ed. IASTED/ACTA Press, 2006, pp. 263–268. [Online]. Available: http://dblp.uni-trier.de/db/conf/iastedCI/ci2006.html#WuNCLW06

BIOGRAPHY

Alain AJAMI is a Ph.D. student at University of Toulon, France. He completed his Master's degree in Engineering (Electrical and control systems) in 2006. In early 2006, he joined the Alcan Engineered Products, a global leader in development and manufacture of high performance and innovative alloys, and was project manager in continuous improvement. In 2010, he started his Ph.D. research in framework of project SHARE on improving the autonomy of UAVs in extensive collaboration with the European aeronautical industry.

Jean-Francois BALMAT is an Assistant Professor at USTV, University of Toulon, France at laboratory LSIS, UMR CNRS 7296. He got Ph.D. degree in Electrical Engineering, Automatic and Signal Processing in 1993. His research areas include identif cation and control by using multi-model approaches.

Jean-Paul GAUTHIER is a professor at USTV, University of Toulon, France, at laboratory LSIS, UMR CNRS 7296. He graduated in maths and computer sciences from Institut National Polytechnique de Grenoble in 1975. He got Ph.D. degree in 1978, and state doctorate degree in 1982. He joined CNRS (National Centre of Scientif c Research) in 1978. He is a member of INRIA team GECO. He is also honorary member of Institut Universitaire de France. His main f eld of interest is control theory in the large.

Thibault MAILLOT is a Ph.D. student at University of Toulon, France. He received his mechatronics Engineer degree from the ENSMM in 2010. In 2010, he started his Ph.D. in control theory, inverse optimal control problem and paths planning in framework of project SHARE.

ANNEXE B. ARTICLE DE CONFÉRENCE CONCERNANT L'APPLICATION DES MÉTHODES DE PLANIFICATION SUR LE SIMULATEUR [?] : *"PATH PLANNING AND GROUND CONTROL STATION SIMULATOR FOR UAV"*

Références

[1] Kenzai Abderrahmane. Planification de trajectoires de robot mobiles via des méthodes ensemblistes. Dea automatique et informatique appliquée, LISA Angers, Juillet 2005.

[2] Ralph Abraham and Jerrold E. Marsden. *Foundations of Mechanics, Second Edition*. Addison-Wesley Publishing Company, Inc., 1987.

[3] Andrei A. Agrachev and Yuri L. Sachkov. *Control theory from the geometric viewpoint*. Encyclopaedia of mathematical sciences. Springer, Berlin, Heidelberg, Paris, 2004. Numérotation dans la collection principale : Encyclopaedia of mathematical sciences, vol. 87.

[4] Alain Ajami, Jean françois Balmat, Jean paul Gauthier, and Thibault Maillot. Path planning and ground control station simulator for uav. In *34th IEEE Aerospace Conference, Montana, USA*, 02 march 2013.

[5] Alain Ajami, Jean-Paul Gauthier, Thibault Maillot, and Ulysse Serres. How humans fly. *ESAIM : Control, Optimisation and Calculus of Variations*, eFirst, 5 2013.

[6] Alain Ajami, Thibault Maillot, Nicolas Boizot, Jean-François Balmat, and Jean-Paul Gauthier. Simulation of a uav ground control station. In *The 9th International Conference of Modeling and Simulation, MOSIM'12*, 2012.

[7] Adel Akbarimajd, Majid Nili Ahmadabadi, and Auke Jan Ijspeert. Analogy between Juggling and Hopping : Active Object Manipulation Approach. *Advanced Robotics*, 25 :1793–1816, 2011.

[8] Andrei A. Ardentov and Yuri L. Sachkov. Inpainting via sub-riemannian minimizers on the group of rototranslations, 2011.

[9] G. Arechavaleta, J-P Laumond, H. Hicheur, and A. Berthoz. Optimizing principles underlying the shape of trajectories in goal oriented locomotion for humans. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 131–136, Dec.

[10] G. Arechavaleta, J-P Laumond, H. Hicheur, and A. Berthoz. The nonholonomic nature of human locomotion : a modeling study. In *Biomedical Robotics and Biomechatronics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*, pages 158–163, Feb. 2006.

[11] Gustavo Arechavaleta, Jean-Paul Laumond, Halim Hicheur, and Alain Berthoz. On the nonholonomic nature of human locomotion. *Autonomous Robots*, 25 :25–35, 2008.

[12] K.B. Ariyur and K.O. Fregene. Autonomous tracking of a ground vehicle by a uav. In *American Control Conference, 2008*, pages 669 –671, june 2008.

[13] E. Bakolas and P. Tsiotras. Time-optimal synthesis for the zermelo-markov-dubins problem : The constant wind case. In *American Control Conference (ACC), 2010*, pages 6163–6168, 30 2010-July 2.

[14] A. Balluchi, A. Bicchi, B. Piccoli, and P. Soueres. Stability and robustness of optimal synthesis for route tracking by dubins' vehicles. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 1, pages 581 –586 vol.1, 2000.

[15] J. Baras, A. Bensoussan, and M. James. Dynamic observers as asymptotic limits of recursive filters : Special cases. *SIAM Journal on Applied Mathematics*, 48(5) :1147–1158, 1988.

[16] A.J. Barry, A. Majumdar, and R. Tedrake. Safety verification of reactive controllers for uav flight in cluttered environments using barrier certificates. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 484–490, 2012.

[17] Térence Bayen, Yacine Chitour, Frédéric Jean, and Paolo Mason. Asymptotic analysis of an optimal control problem connected to the human locomotion. 48th IEEE Conference on Decision and Control, Shangai, 2009.

[18] Calin Belta, Antonio Bicchi, Magnus B. Egerstedt, Emilio Frazzoli, Eric Klavins, and George J. Pappas. Symbolic planning and control of robot motion : State of the art and grand challenges. *IEEE Robotics and Automation Magazine*, Vol. 14, March 2007.

[19] Bastien Berret. *Intégration de la force gravitaire dans la planification motrice et le contrôle des mouvements du bras et du corps.* Applied mathematics, Université de Bourgogne, 2008.

[20] Bastien Berret, Christian Darlot, Frédéric Jean, Thierry Pozzo, Charalambos Papaxanthis, and Jean Paul Gauthier. The inactivation principle : mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements. *PLoS Comput. Biol.*, 4(10) :e1000194, 25, 2008.

[21] S. Bertrand, T. Hamel, H. Piet-Lahanier, and R. Mahony. Attitude tracking of rigid bodies on the special orthogonal group with bounded partial state feedback. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 2972 –2977, dec. 2009.

[22] L.F. Bertuccelli, A. Wu, and J.P. How. Robust adaptive markov decision processes : Planning with model uncertainty. *Control Systems, IEEE*, 32(5) :96–109, 2012.

[23] G. Besancon. *Nonlinear Observers and Applications.* Lecture notes in control and information sciences. Springer London, Limited, 2007.

[24] A. Bhatia and E. Frazzoli. Decentralized algorithm for minimum-time rendezvous of dubins vehicles. In *American Control Conference, 2008*, pages 1343–1349, June.

[25] A. Bhatia, M. Graziano, S. Karaman, R. Naldi, and E. Frazzoli. Dubins trajectory tracking using commercial off-the-shelf autopilots. In *AIAA Guidance, Navigation, and Control Conference*, Honolulu, Hawaii, August 2008.

[26] J.L. Boiffier. *The dynamics of flight : the equations.* Dynamics of Flight Series. Wiley, 1998.

[27] Jean-Daniel Boissonnat, André Cérézo, and Juliette Leblond. Shortest paths of bounded curvature in the plane. *Journal of Intelligent and Robotic Systems*, 11 :5–20, 1994.

[28] B. Bonnard, V. Jurdjevic, I. Kupka, and G. Sallet. Transitivity of families of invariant vector fields on the semi-direct product of lie groups. *Trans.Amer.Math.Soc.*, 271(2) :525–535, 1982.

[29] Guy Bornard and Jean-Paul Gauthier. Commande dynamique multivariable des systèmes industriels de production. Note interne LAG N.77-29, Institut National Polytechnique de Grenoble, Novembre 1977.

[30] U. Boscain, G. Charlot, and F. Rossi. Existence of planar curves minimizing length and curvature. *ArXiv e-prints*, June 2009.

[31] U. Boscain and B. Piccoli. *Optimal Syntheses for Control Systems on 2-D Manifolds*. Mathématiques et Applications. Springer, 2004.

[32] Ugo Boscain and Benedetto Piccoli. Extremal synthesis for generic planar systems. *Journal of Dynamical and Control Systems*, 7(2) :209–258, April 2001.

[33] A. Bressan and B. Piccoli. A generic classification of time-optimal planar stabilizing feedbacks. *SIAM J. Control Optim.*, 36(1) :12–32 (electronic), 1998.

[34] Xuân-Nam Bui, Philippe Soueres, Jean-Daniel Boissonnat, and Jean-Paul Laumond. The Shortest path synthesis for non-holonomic robots moving forwards. Rapport de recherche RR-2153, INRIA, 1994.

[35] Francesco Bullo and Andrew D. Lewis. *Geometric Control of Mechanical Systems*, volume 49 of *Texts in Applied Mathematics*. Springer Verlag, New York-Heidelberg-Berlin, 2004.

[36] D. Campolo, L. Schenato, E. Guglielmelli, and S. Sastry. A lyapunov-based approach for the control of biomimetic robotic systems with periodic forcing inputs. In *Proceedings of 16th IFAC World Congress on Automatic Control (IFAC05)*, 2005.

[37] HongDa Chen, KuoChu Chang, and C.S. Agate. Tracking with uav using tangent-plus-lyapunov vector field guidance. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 363 –372, july 2009.

[38] Yacine Chitour, Frédéric Jean, and Ruixing Long. A Global Steering Method for Nonholonomic Systems. grant from Digiteo and Région Ile-de-France.

[39] Yacine Chitour, Frédéric Jean, and Paolo Mason. Optimal control models of the goal-oriented human locomotion. *SIAM J. Control and Optimization*, 50, No.1, 1 2012.

[40] H. Chitsaz and S.M. LaValle. Time-optimal paths for a dubins airplane. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2379 –2384, dec. 2007.

[41] Francesca Chittaro, Frédéric Jean, and Paolo Mason. On the inverse optimal control problems of the human locomotion : stability and robustness of the minimizers. 20 pages, 2011.

[42] Chee-Yee Chong, D. Garren, and T.P. Grayson. Ground target tracking-a historical perspective. In *Aerospace Conference Proceedings, 2000 IEEE*, volume 3, pages 433 –448 vol.3, 2000.

[43] F.H. Clarke. *Optimization and Nonsmooth Analysis*. Classics in applied mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1990.

[44] Jean-Michel Coron. Quelques résultats sur la commandabilité et la stabilisation des systèmes non linéaires. Jounées X-UPS, 1999.

[45] Jean-Michel Coron. On the controllability of nonlinear partial differential equations. In *Proceedings of the International Congress of Mathematicians Hyderabad*, pages p. 239–264. World Scientific Publishing Co Pte Ltd, 2010. Vol. I : Plenary Lectures and Ceremonies.

[46] M.L. Cummings, J.T. Platts, and A. Sulmistras. Human performance considerations in the development of interoperability standards for uav interfaces. In *Moving Autonomy Forward Conference*, 2006.

[47] Dimos V. Dimarogonas, Savvas G. Loizou, Kostas J. Kyriakopoulos, and Michael M. Zavlanos. A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. *Automatica*, 42(2) :229 – 243, 2006.

[48] Xu Chu Ding, A.R. Rahmani, and M. Egerstedt. Multi-uav convoy protection : An optimal approach to path planning and coordination. *Robotics, IEEE Transactions on*, 26(2) :256 –268, april 2010.

[49] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3) :197–208, 2000.

[50] Mikalai Drabovich. Automated mission trajectory planning for mine countermeasures operations. Master's thesis, The School of Engineering and Physical Sciences Heriot Watt University, Edinburgh, 2008.

[51] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79 :497–516, 1957.

[52] Tom Erez and Emanuel Todorov. Trajectory optimization for domains with contacts using inverse dynamics. In *IROS*, pages 4914–4919, 2012.

[53] M. Fliess, J. Lévine, Ph. Martin, and P. Rouchon. Design of trajectory stabilizing feedback for driftless flat systems. *Proc. of the 3-rd European Control Conf*, pages 1882–1887, 1995.

[54] Michel Fliess. Variations sur la notion de contrôlabilité. 2000.

[55] Michel Fliess, Jean Lévine, and Pierre Rouchon. Flatness and defect of nonlinear systems : Introductory theory and examples. *International Journal of Control*, 61 :1327–1361, 1995.

[56] FLIR Systems, PORTLAND, OREGON USA. *Star SAFIRE HD Gen II Product Specifications*, 2010. Document No : 4115214 Rev 0.

[57] Robin Forman. Witten–Morse theory for cell complexes. *Topology*, 37(5) :945–979, March 1998.

[58] E. Frazzoli, M. A. Dahleh, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Trans. on Robotics*, 21(6) :1077–1091, December 2005.

[59] E. Frew, T. McGee, ZuWhan Kim, Xiao Xiao, S. Jackson, M. Morimoto, S. Rathinam, J. Padial, and R. Sengupta. Vision-based road-following using a small autonomous aircraft. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 5, pages 3006 – 3015 Vol.5, march 2004.

[60] Eric W Frew, Dale A Lawrence, and Steve Morris. Coordinated standoff tracking of moving targets using lyapunov guidance vector fields. *Journal of Guidance Control and Dynamics*, 31(2) :290–306, 2008.

[61] Jean-Paul Gauthier and Ivan Kupka. *Deterministic observation theory and applications*. Cambridge University Press, Cambridge, 2001.

[62] V. Gavrilets, B. Mettler, and E. Feron. Nonlinear model for a small-size acrobatic helicopter. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2001.

[63] T. Hamel and R. Mahony. Visual servoing of an under-actuated dynamic rigid-body system : an image-based approach. *Robotics and Automation, IEEE Transactions on*, 18(2) :187–198, Apr 2002.

[64] John A. Hancock. *Laser Intensity-Based Obstacle Detection and Tracking*. PhD thesis, The Robotics Institute, Carnegie Mellon University, 1999.

[65] Nalin Harischandra, Jeremei Knuesel, Alexander Kozlov, Andrej Bicanski, Jean-Marie Cabelguen, Auke Jan Ijspeert, and Örjan Ekeberg. Sensory feedback plays a significant role in generating walking gait and in gait transition in salamanders : A simulation study. *Frontiers in Neurorobotics*, 5(3), 2011.

[66] P. E. Hart, N. J. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2) :100–107, July 1968.

[67] Helmut Hauser, Auke J. Ijspeert, Rudolf M. Fuechslin, Rolf Pfeifer, and Wolfgang Maass. Towards a theoretical foundation for morphological computation with compliant bodies. *Biological Cybernetics*, 105 :355–370, 2011.

[68] Kris Hauser, Tim Bretl, Kensuke Harada, and Jean claude Latombe. Using motion primitives in probabilistic sample-based planning for humanoid robots. In *In WAFR*, 2006.

[69] Gabriel M. Hoffmann, Steven L. Waslander, and Claire J. Tomlin. Quadrotor helicopter trajectory tracking control. In *In Proc. AIAA Guidance, Navigation, and Control Conf*, 2008.

[70] Richard G.B. Howson and al. System and methods for autonomous tracking and surveillance, 2008.

[71] Minh-Duc Hua, T. Hamel, P. Morin, and C. Samson. A control approach for thrust-propelled underactuated vehicles and its application to vtol drones. *Automatic Control, IEEE Transactions on*, 54(8) :1837 –1853, aug. 2009.

[72] Craig R. Husby. Path generation tactics for a uav following a moving target. Master's thesis, University of Washington, 2005.

[73] Jean-Michel Hélary. Algorithmiques des graphes. Cours, Juin 2004. IFSIC.

[74] ICAO. Unmanned aircraft systems (uas), 2011.

[75] Frederic Jean, Ruixing Long, Giuseppe Oriolo, and Marilena Vendittelli. An approximate algorithm for nonholonomic motion planning. Technical report, Ecole polytechnique, CMAP, 2008.

[76] H.K. Khalil. *Nonlinear systems*. Macmillan Pub. Co., 2002.

[77] H.J. Kim, D.H. Shim, and S. Sastry. Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. In *American Control Conference, 2002. Proceedings of the 2002*, volume 5, pages 3576 – 3581 vol.5, 2002.

[78] Jongrae Kim and Yoonsoo Kim. Moving ground target tracking in dense obstacle areas using uavs. In Misra Pradeep Chung, Myung Jin, editor, *Proceedings of the 17th IFAC World Congress*, pages 8552–8557, 2008.

[79] Z.W. Kim. Realtime road detection by learning from one example. In *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on*, volume 1, pages 455 –460, jan. 2005.

[80] J. P. Lasalle. Stability theory for ordinary differential equations. *Journal of Differential Equations*, 4 :57–65, 1968.

[81] Jean-Claude Latombe. A fast path planner for a car-like indoor mobile robot. In *Proceedings of the ninth National conference on Artificial intelligence - Volume 2*, AAAI'91, pages 659–665. AAAI Press, 1991.

[82] J.P. Laumond. *Robot motion planning and control*. Lecture notes in control and information sciences. Springer, 1998.

[83] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http ://planning.cs.uiuc.edu/.

[84] Dale A Lawrence, Eric W Frew, and William J Pisano. Lyapunov vector fields for autonomous unmanned aircraft flight control. *Journal of Guidance Control and Dynamics*, 31(5) :1220–1229, 2008.

[85] Jusuk Lee, Rosemary Huang, Andrew Vaughn, Xiao Xiao, J Karl Hedrick, Marco Zennaro, and Raja Sengupta. Strategies of path-planning for a uav to track a ground vehicle. *AINS Conference*, 2003.

[86] Francois LeGland. Filtrage particulaire. Rapport de laboratoire IRISA/INRIA, 2011.

[87] Weiwei Li, Emanuel Todorov, and Dan Liu. Inverse optimality design for biological movement systems. In *World Congress, Volume 18 Part 1*, 2011.

[88] Ruixing Long. *Planification de mouvements pour les systèmes non-holonomes et étude de la contrôlabilité spectrale pour les équations de Schrödinger linéarisées*. PhD thesis, Ecole Polytechnique X, 2010.

[89] Robert Mahony and Tarek Hamel. Robust trajectory tracking for a scale model autonomous helicopter. *International Journal of Robust and Nonlinear Control*, 14(12) :1035–1059, 2004.

[90] Philippe Martin, Pierre Rouchon, Jean-Michel Coron, and Jean-Pierre Puel. *Aspect de la théorie du contrôle*, volume Journées mathématiques X-UPS 1999. éditions de l'école polytechnique edition, 1999.

[91] Ivan Maza and Anibal Ollero. Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In Rachid Alami, Raja Chatila, and Hajime Asama, editors, *Distributed Autonomous Robotic Systems 6*, pages 221–230. Springer Japan, 2007.

[92] Igor Moiseev and Yuri L. Sachkov. Maxwell strata in sub-Riemannian problem on the group of motions of a plane. *ESAIM Control Optim. Calc. Var.*, 16(2) :380–399, 2010.

[93] Vincent Morio. *Contribution au développement d'une loi de guidage autonome par platitude. Application à une mission de rentrée atmosphérique.* These, Université Sciences et Technologies - Bordeaux I, May 2009.

[94] Stephen Morris and Eric W. Frew. Cooperative tracking of moving targets by teams of autonomous unmanned air vehicles. Final report, University of Colorado at Boulder, MLB Company, 2551 Casey Ave, Suite B, Mountain View, CA 94043, July 2005.

[95] NATO. *STANAG 4586 - B - 80*, 2008.

[96] Frank Nielsen, Jean-Daniel Boissonnat, and Richard Nock. Bregman voronoi diagrams : Properties, algorithms and applications. *CoRR*, abs/0709.2196, 2007.

[97] Office of the Secretary of Defense. Unmanned aircraft systems roadmap 2005-2030. Technical report, Departement of Defense USA, 2005.

[98] Andres Ortiz, Derek Kingston, and Cédric Langbort. Multi-uav velocity and trajectory scheduling strategies for target classification by a single human operator. *Journal of Intelligent & Robotic Systems*, 70 :255–274, 2013.

[99] Sofiane Ouanezar, Selim Eskiizmirliler, Frédéric Jean, Bertrand Tondu, Marc Maier, and Christian Darlot. Biologically inspired sensory motor control of a 2-link robotic arm actuated by mckibben muscles. 2011 IEEE International Conference on Robotics and Automation, 2011.

[100] Sanghyuk Park, John Deyst, and Jonathan P. How. A new nonlinear guidance logic for trajectory tracking. In *In Proceedings of the AIAA Guidance, Navigation and Control Conference*, pages 2004–4900, 2004.

[101] M. Perrollaz, R. Labayrade, C. Royere, N. Hautiere, and D. Aubert. Long range obstacle detection using laser scanner and stereovision. In *Intelligent Vehicles Symposium, 2006 IEEE*, pages 182–187, 2006.

[102] Nicolas Petit and Pierre Rouchon. Automatique : Dynamique et contrôle des systèmes. December 2009.

[103] B. Piccoli. Classification of generic singularities for the planar time-optimal synthesis. *SIAM J. Control Optim.*, 34(6) :1914–1946, November 1996.

[104] Benedetto Piccoli. Optimal syntheses for state constrained problems with application to optimization of cancer therapies. *Mathematical Control and Related Fields*, 2(4) :383–398, 2012.

[105] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition : The Art of Scientific Computing.* Cambridge University Press, New York, NY, USA, 3 edition, 2007.

[106] Carole Gabrielle Prévost, Andre Desbiens, and Daniel Gagnon, Ericand Hodouin. Uav optimal cooperative target tracking and collision avoidance of moving objects. In Chung, Myung Jin, Misra, and Pradeep, editors, *The International Federation of Automatic Control, Vol. 17*, pages 5724–5729, 2008.

[107] F. Rafi, S. Khan, K. Shafiq, and M. Shah. Autonomous target following by unmanned aerial vehicles. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 6230 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, June 2006.

[108] S. Rathinam, P. Almeida, ZuWhan Kim, S. Jackson, A. Tinka, W. Grossman, and R. Sengupta. Autonomous searching and tracking of a river using an uav. In *American Control Conference, 2007. ACC '07*, pages 359 –364, july 2007.

[109] S. Rathinam, Zu Kim, A. Soghikian, and R. Sengupta. Vision based following of locally linear structures using an unmanned aerial vehicle. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, pages 6085 – 6090, dec. 2005.

[110] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2) :367–393, 1990.

[111] Frédéric Rotella and Irène Zambettakis. Commande des systèmes par platitude. *Techniques de l'ingénieur Régulation et commande des systèmes asservis*, base documentaire : TIB394DUO.(ref. article : s7450), 2007. fre.

[112] Frédéric L. Roubieu, Julien Serres, Nicolas Franceschini, Franck Ruffier, and Stéphane Viollet. A fully-autonomous hovercraft inspired by bees : wall following and speed control in straight and tapered corridors.

[113] Pierre Rouchon. Contrôle des systèmes non-linéaires. Tunis, Novembre 2011. Slides de cours.

[114] Franck Ruffier, Simon Benachio, Fabien Expert, and Erick Ogam. A tiny directional sound sensor inspired by crickets designed for micro-air vehicles. In *IEEE Sensors 2011 conference*, page CDROM, Limerick, Irlande, 2011. PIR.

[115] Yu.L. Sachkov. Control theory on lie groups. *Journal of Mathematical Sciences*, 156 :381–439, 2009.

[116] Yuri L. Sachkov. Conjugate and cut time in the sub-Riemannian problem on the group of motions of a plane. *ESAIM Control Optim. Calc. Var.*, 16(4) :1018–1039, 2010.

[117] Yuri L. Sachkov. Cut locus and optimal synthesis in the sub-Riemannian problem on the group of motions of a plane. *ESAIM Control Optim. Calc. Var.*, 17(2) :293–321, 2011.

[118] Jean-Christophe Sarrazin, Arnaud Tonnelier, and Frederic Alexandre. A model of contextual effect on reproduced extents in recall tasks : the issue of the imputed motion hypothesis. *Biological Cybernetics*, 92 :303–315, 2005.

[119] A. Sarti, G. Walsh, and S. Sastry. Steering left-invariant control systems on matrix lie groups. In *Decision and Control, 1993., Proceedings of the 32nd IEEE Conference on*, pages 3117 –3121 vol.4, dec 1993.

[120] K. D. Sebesta and N. Boizot. Real-time adaptive high-gain ekf, applied to a quadcopter inertial navigation system. *IEEE Trans. on Indus. Elec.*, To appear.

[121] Kenneth Sebesta. *Optimal Observers and Optimal Control : Improving Car Efficiency with Kalman and Pontryagin.* PhD thesis, University of Luxembourg, 2010.

[122] P. Soueres and J.-P. Laumond. Shortest paths synthesis for a car-like robot. *Automatic Control, IEEE Transactions on*, 41(5) :672 –688, may 1996.

[123] Army UAS CoE Staff. "eyes of the army" u.s. army unmanned aircraft systems roadmap 2010-2035. Technical report, U.S. ARMY, 2010.

[124] S. Sugimoto, H. Tateda, H. Takahashi, and M. Okutomi. Obstacle detection using millimeter-wave radar and its visualization on image sequence. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 342–345 Vol.3, 2004.

[125] H. Sussmann. Regular synthesis for time-optimal control of single-input real analytic systems in the plane. *SIAM Journal on Control and Optimization*, 25(5) :1145–1162, 1987.

[126] Héctor J. Sussmann and Velimir Jurdjevic. Controllability of nonlinear systems. *J. Differential Equations*, 12 :95–116, 1972.

[127] Héctor J. Sussmann and Guoqing Tang. Shortest paths for the reeds-shepp car : A worked out example of the use of geometric techniques in nonlinear optimal control. Technical report, 1991.

[128] Ryo Takei and Richard Tsai. Optimal trajectories of curvature constrained motion in the hamilton–jacobi formulation. *Journal of Scientific Computing*, 54 :622–644, 2013.

[129] Tertia. Estimation non linéaire optimale. filtrage particulaire. Rapport d'activité DIGINEXT, Octobre 1996.

[130] P. Theodorakopoulos and S. Lacroix. A strategy for tracking a ground target with a uav. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1254 –1259, sept. 2008.

[131] Justin Thomas, Joe Polin, Koushil Sreenath, and Vijay Kumar. Avian-inspired grasping for quadrotor micro uavs. In *ASME International Design Engineering Technical Conference (IDETC)*, submitted, 2013.

[132] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2) :99–141, 2000.

[133] J. Tisdale, ZuWhan Kim, and J.K. Hedrick. Autonomous uav path planning and estimation. *Robotics Automation Magazine, IEEE*, 16(2) :35–42, 2009.

[134] Emanuel Todorov. Optimality principles in sensorimotor control. *Nature Neuroscience*, (9) :907–915, 2004.

[135] Emmanuel Trélat. *Contrôle optimal : théorie & applications*. Mathématiques concrètes. Vuibert, Paris, 2005.

[136] J.N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *Automatic Control, IEEE Transactions on*, 40(9) :1528 –1538, sep 1995.

[137] M. van Manen and D. Siersma. Power diagrams and their applications. *ArXiv Mathematics e-prints*, August 2005.

[138] Michiel van Nieuwstadt and Richard M. Murray. Real time trajectory generation for differentially flat systems. *Int. Journal of Robust and Nonlinear Control*, 8 :995–1020, 1996.

[139] Richard Vinter. *Optimal control*. Systems & Control : Foundations & Applications. Birkhäuser Boston Inc., Boston, MA, 2000.

[140] Michael Vitus, Vijay Pradeep, Gabriel M. Hoffmann, Steven L. Waslander, and Claire J. Tomlin. Tunnel-MILP : Path planning with sequential convex polytopes. In *2008 AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, USA, August 2008.

[141] G.C. Walsh, R. Montgomery, and S.S. Sastry. Optimal path planning on matrix lie groups. In *Decision and Control, 1994., Proceedings of the 33rd IEEE Conference on*, volume 2, pages 1258 –1263 vol.2, dec 1994.

[142] Rohin Wood. *Lyapunov-based Control Strategies for the Global Control of Symmetric VTOL UAVs*. PhD thesis, School of Mechanical Engineering, The University of Adelaide, South Australia 5005, Australia, June 2007. Thesis (Ph.D.) – University of Adelaide, School of Mechanical Engineering, 2007.

[143] E. Xargay, V. Dobrokhodov, I. Kaminer, A.M. Pascoal, N. Hovakimyan, and Chengyu Cao. Time-critical cooperative control of multiple autonomous vehicles : Robust distributed strategies for path-following control and time-coordination over dynamic communications networks. *Control Systems, IEEE*, 32(5) :49–73, 2012.

[144] Senqiang Zhu, Danwei Wang, and ChangBoon Low. Ground target tracking using uav with input constraints. *Journal of Intelligent & Robotic Systems*, 69(1-4) :417–429, 2013.

Title : Path planning of unmanned combat aircraft vehicles

Abstract : This thesis is about path planning for HALE or MALE UAVs (Unmanned Aircraft Vehicles), possibly under mission constraints. As such, the study is performed at the kinematic level : HALE UAVs are represented as Dubins systems, and a model for MALE UAVs is constructed by studying their kinematic frame.

In the first part, we tackle the path planning problem for a UAV that must join a target (a point or a pattern), starting from any position. The point to point path planning problem is addressed as an optimal control problem. Regarding the point to pattern path planning problem, two different methods are proposed. The former consists in solving the minimum time synthesis for the Dubins system, in order to obtain a basis for a HALE UAVs planning algorithm. The latter method relies on the LaSalle principle ; it permits to stabilize a HALE or MALE UAV to a pattern.

In addition, extensions of the previously developed algorithms to cluttered environnement are provided. This extension is achieved thanks to a space discretization using Voronoi diagrams and a discrete planning method.

Finally, the mission constraints are dealt with as a coupling problem between the UAV and its sensors. The proposed algorithm is presented in the form of a constrained quadratic problem.

In the second part of this thesis, we want to refine the planning algorithm to get a result closer to trajectories of pilots. In order to do that, we solve an inverse optimal control problem where the cost to find is computed from the experience of pilots. Theoretical results are presented and applied to the particular case of the Dubins system.

Keywords : Optimal control, Lyapunov method, LaSalle principle, inverse optimal control problem, anthropomorphic control, drone, identification, modeling, path planning.

Titre : Planification de trajectoire pour drones de combat

Résumé : L'objectif principal de ce travail est l'étude de la planification de trajectoires pour des drones de type HALE ou MALE. Les modèles cinématiques de ces drones sont étudiés. Les drones HALE sont modélisés par le système de Dubins. Pour les drones MALE, le modèle est construit en étudiant le repère cinématique du drone.

Nous considérons les problèmes de planification de trajectoires point-point et point-pattern. Il s'agit, à partir de la position courante du drone, de rejoindre un point ou une figure prédéfinie dans l'espace. La planification point-point est abordée sous forme d'un problème de contrôle optimal. Deux méthodes sont proposées pour résoudre le problème point-pattern. D'abord nous présentons la synthèse en temps minimal pour le système de Dubins. Ensuite, nous développons une méthode basée sur le principe de LaSalle. La première méthode est utilisée au sein d'un algorithme de planification pour des drones HALE. La deuxième permet de stabiliser les deux types de drones considérés vers un pattern.

Nous proposons une extension des algorithmes de planification développés, basée sur une discrétisation de l'espace grâce aux graphes de Voronoï et une méthode de planification discrète, pour construire des trajectoires dans des milieux encombrés.

Nous étudions également le problème de couplage drone/capteur. Il s'agit de calculer une trajectoire permettant de satisfaire les objectifs du drone et de son capteur (une caméra). L'algorithme proposé est construit à partir de la résolution d'un problème quadratique sous contraintes.

Dans une seconde partie, nous analysons un problème de contrôle optimal inverse. Celui-ci permet d'améliorer les résultats des méthodes de planification en s'inspirant du comportement des pilotes. Après avoir posé le problème, les résultats théoriques sont exposés et le cas particulier du système de Dubins est étudié en pratique.

Mots-clefs : Contrôle optimal, méthode de Lyapunov, principe de LaSalle, problème de contrôle optimal inverse, commande anthropomorphique, drone, identification, modélisation, planification de trajectoires.